UNIVERSIDADE DE LISBOA

Faculdade de Ciências

Departamento de Informática

WEBSPOTS

SUBSCRIÇÃO DE SERVIÇOS VIA WEB

Bruno Filipe Lourenço Coelho

Mestrado em Engenharia Informatica

Especialização: Sistemas de Informação

2008

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

WEBSPOTS

SUBSCRIÇÃO DE SERVIÇOS VIA WEB

Bruno Filipe Lourenço Coelho

PROJECTO

Projecto orientado pela Prof. Dr. Teresa Chambel
e co-orientado por João Paulo Pereira

Mestrado em Engenharia Informática

Especialização: Sistemas de Informação

2008

DECLARAÇÃO

Bruno Filipe Lourenço Coelho, aluno nº 32040 da Faculdade de Ciências da Universidade de Lisboa, declara ceder os seus direitos de cópia sobre o seu Relatório de Projecto em Engenharia Informática, intitulado *"*Webspots, subscrição de serviços via web*",* realizado no ano lectivo de 2007/2008 à Faculdade de Ciências da Universidade de Lisboa para o efeito de arquivo e consulta nas suas bibliotecas e publicação do mesmo em formato electrónico na Internet.

FCUL, 20 de Agosto de 2008

Dr. João Paulo Fernandes Pereira, supervisor do projecto de Bruno Filipe Lourenço Coelho, aluno da Faculdade de Ciências da Universidade de Lisboa, declara concordar com a divulgação do Relatório do Projecto em Engenharia Informática, intitulado "Webspots, subscrição de serviços via web ".

Lisboa, 20 de Agosto de 2008

RESUMO

A TIMw.e. apresenta-se como uma empresa de distribuição de conteúdos para dispositivos móveis. Estes conteúdos são fornecidos de várias formas, sendo que as mais populares se identificam com processos de subscrição e fidelização de clientes.

Existem diversos meios de publicitar estes conteúdos, um deles foi objecto de estudo e desenvolvido neste trabalho e é descrito neste relatório. Os *Webspots* são os mecanismos de publicidade mais populares e dinâmicos utilizados pela empresa.

O trabalho realizado consistiu na realização de uma *framework* para desenvolvimento de *Webspots* para os Estados Unidos da América. Esta *framework* permite disponibilizar os mecanismos necessários para a publicidade e subscrição de serviços, tendo em conta as particularidades do pais em questão, nomeadamente interacções com operadores e brokers específicas desse país.

Este trabalho insere-se no âmbito do Projecto em Engenharia Informática, do Mestrado em Engenharia Informática da Faculdade de Ciências da Universidade de Lisboa.

Palavras-chave: Publicidade; Distribuição; Conteúdos; Telemóvel; Webspot;

ABSTRACT

TIMw.e. presents itself as a mobile content distributon company. These contents are supplied in various ways, being the most popular, the ones related to client subscription model.

There are lots of different ways of advertising these contents, one of those, was the object of a study and its development is described in the following report. Webspots are the most popular type of advertisement TIMw.e. uses.

The work described on the report consisted on the implementation of a framework that allowed the development of Webspots for the United States of America. This framework

allows the clustering of mechanisms for the advertisement of services, taking in to consideration the particularities of the country (USA), mainly interaction with the broker and the operators.

Keywords: Advertising; Distribution; Content; Phone; Webspot

Agradecimentos:

Gostaria de agradecer a todos os que me permitiram desenvolver este projecto, todo o apoio que me deram.

Gostaria também de agradecer a professora Teresa Chambel, pelos conselhos e criticas que me fez, por forma a melhorar o relatório que desenvolvi.

Índice

Índice de Imagens

Índice de Tabelas

1. Introdução

Webspot é uma ferramenta que faz a publicidade de conteúdos na Web, desenvolvida pela empresa TIMw.e. (TIM wireless entertainment) para distribuição de conteúdos em dispositivos móveis. Esta ferramenta permite uma melhor racionalização de recursos por parte da empresa, e um acesso a conteúdos mais eficiente por parte do consumidor, pois publicita apenas certos tipos de conteúdos. A título de exemplo podemos considerar um *Webspot* da Shakira, cujo interesse é orientar os clientes única e exclusivamente para conteúdos relativos à artista em questão (quer sejam jogos, *wallpapers, screensavers*, músicas, entre muitos outros). Este *Webspot* permite que o cliente, após a inserção do seu número de telefone e a correspondente *password*, fique registado no clube da Shakira e possa receber os conteúdos disponibilizados por esse mesmo clube.

Em conjunto com a publicidade na TV, nas revistas, entre outras, os *Webspots* permitem angariar clientes para os serviços disponibilizados pela empresa (*one-shots*, clubes de subscrição, *wapsites*, etc...). Na figura 1 encontra-se, a título de exemplo, a interface de um *Webspot* criado para o mercado português:

Figura 1: Layout genérico de um *Webspot*

1.1.Motivação

Relativamente à inserção do projecto no âmbito do negócio da empresa, este demarca um dos mais bem-sucedidos mecanismos de distribuição de conteúdos pela Internet, por parte da empresa TIMw.e.. Pensado e criado pela mesma, veio responder às necessidades cada vez mais especificas dos clientes, possibilitando-lhes um redireccionamento directo para os conteúdos pretendidos. Os *Webspots* são sem dúvida um dos mais populares, aparte de outros meios de distribuição de conteúdos, sendo que, para além de uma grande flexibilidade no que toca a aglomeração de conteúdos, permitem uma boa adaptação às mentalidades e padrões existentes nos mercados alvo. Recorrendo ao exemplo dos Estados Unidos, o mercado responde mal a certas especificidades, como por exemplo: caixa única de inserção do número de telemóvel; lista com operadores disponíveis, visível no ecrã, entre outras.

A *framework* desenvolvida neste projecto acompanha o lançamento do projecto do país em questão: Estados Unidos da América. As *frameworks* são criadas após as ligações com os operadores estarem concluídas, e após existirem clubes de subscrição, ou outros serviços, criados para aglomeração de conteúdos. Permitem uma rápida criação de *Webspots* com os padrões necessários para esse país, sendo que, após a sua conclusão, apenas se torna necessária a manipulação de ficheiros de propriedades para criação dos mais diversos *Webspots.*

1.2.Objectivos

Este projecto teve como objectivo a realização de uma *framework* para construção de *Webspots* com os requisitos necessários (por exemplo, em termos de interacções com *brokers* e/ou operadores) para o mercado dos Estados Unidos da América, bem como na concretização de um *Webspot* para distribuição de conteúdos de um clube de subscrição (que quando subscrito disponibiliza ao cliente um variado numero de conteúdos). Após a implementação da *framework*, a criação de *Webspots* poderá ser feita com maior facilidade e apenas recorrendo a pequenas configurações em termos de interface, relativamente as principais funcionalidades a concretizar, encontra-se um serviço para obtenção do operador de um número introduzido no *Webspot*, um mecanismo de verificação para números bloqueados, mecanismos de lembrança para as mensagens de *password* (mensagens que contêm a *password* do cliente) enviadas, entre outras. Estas

funcionalidades serão suportadas pela *framework* de modo a estarem presentes em todos os *Webspots*. O acesso ao *Webspot* irá ser feito posteriormente via Web, quer por pesquisa directa, quer por *adsenses*.

1.3.Estrutura do Documento

O documento encontra-se estruturado da seguinte forma:

- Capítulo 2:
  - Apresentação da empresa;
  - Identificação dos tipos de negócio praticados;
  - Conteúdos disponibilizados pela empresa;
  - Identificação e descrição dos serviços existentes;
  - Integração na empresa;
- Capítulo 3:
  - Planeamento do projecto;
- Capítulo 4:
  - Definição do conceito de *Webspot*;
  - Versões, arquitecturas, funcionalidades de um *Webspot*;
- Capítulo 5:
  - Descrição do trabalho desenvolvido (desenho, implementação e testes);
- Capítulo 6:
  - Conclusão sobre o trabalho desenvolvido.

2. Empresa

A TIMw.e. apresenta-se primordialmente como uma empresa de distribuição de conteúdos para dispositivos móveis. Encontra-se com negócios em cerca de 60 países, tendo vindo a crescer de ano para ano. Com a sede localizada em Portugal, a empresa edificada no ano de 2002 possui escritórios em vários outros países (18). As equipas de trabalho encontram-se divididas por país, sendo que numa primeira instância o lançamento do projecto de um país é feito em Portugal, e após a sua implementação (se for melhor para o negocio) é migrado para o pais em questão.

2.1.Tipo de Negócio

A empresa suporta dois principais tipos de negócio:

- B2C (*Business to Client*) – tem como objectivo a distribuição de conteúdos para um cliente final, como é apresentado de seguida:

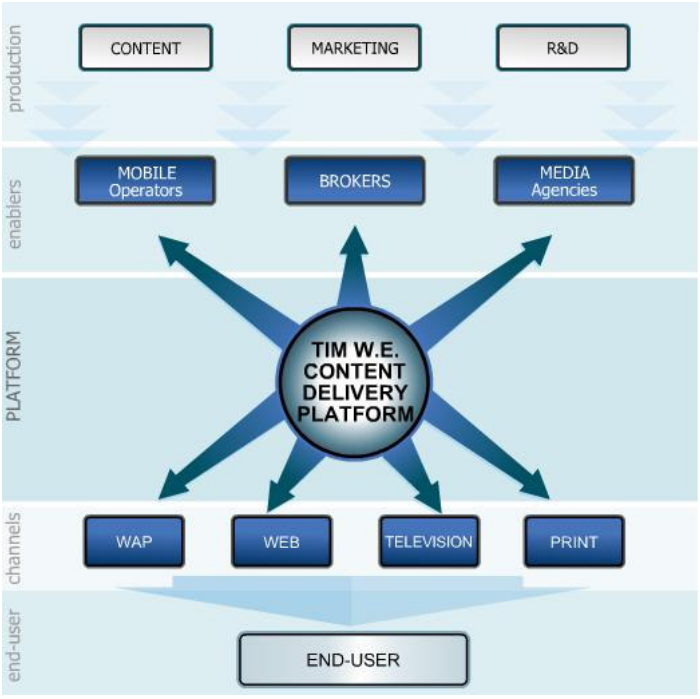


Figura 2 : Diagrama de negócio Business to Client

- B2B (*Business to Business*) – permite fornecer uma plataforma de distribuição de conteúdos, tendo como clientes outras empresas:

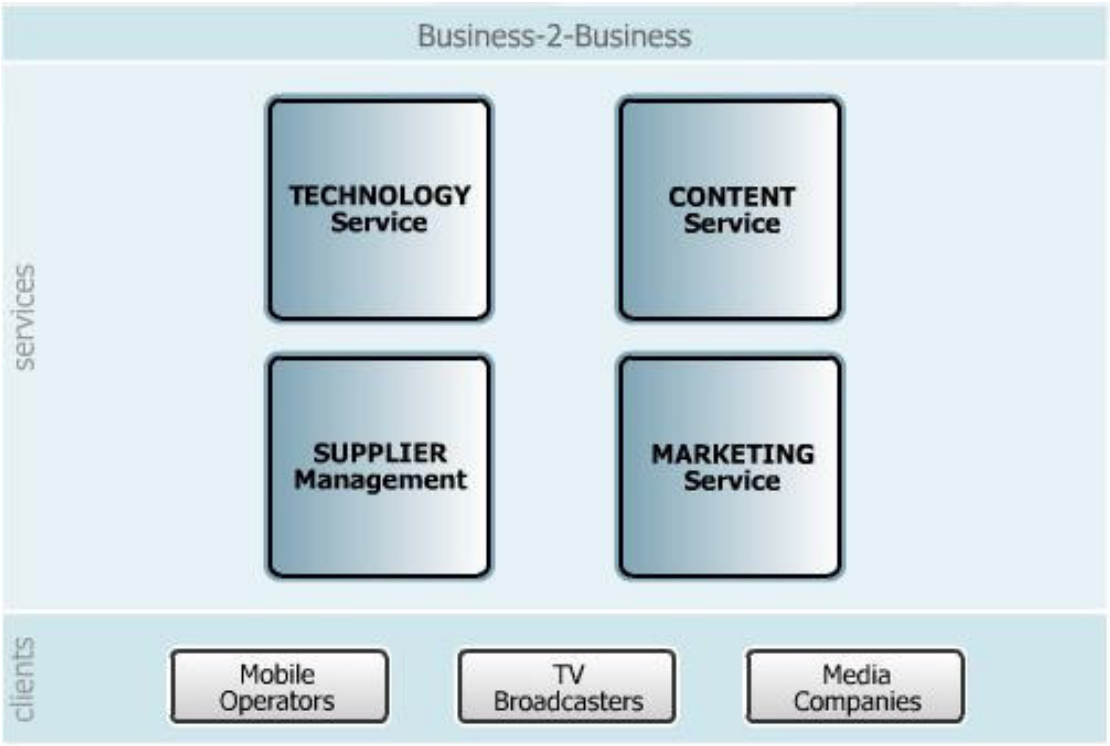


Figura 3 : Diagrama de negócio Business to Business

Estes tipos de negócio permitem que a empresa possa integrar-se de diferentes maneiras no mercado, gerando ideias próprias de negócio ou simplesmente implementando ideias que os parceiros fornecem.

A plataforma usada pela empresa assenta em sistemas Linux e Solaris, que correm aplicações J2EE sobre Oracle. Esta plataforma possui módulos especializados em diferentes vertentes do negócio gerido pela empresa. A *gateway* SMS é um dos maiores componentes, permitindo a existência de conexões com cerca de 120 operadores a nível mundial, usando 27 diferentes tipos de protocolos e tornando possível a independência de conexões, filas de mensagens, compromisso de entrega de mensagens, e mecanismos de *reporting* (consulta de dados estatísticos). Esta *gateway* suporta 200 milhões de SMS por mês.

Em seguida mostramos as particularidades da plataforma utilizada, onde podemos encontrar entre outras as *gateways* responsáveis pelo envio e recepção de mensagens, acesso a *WAP sites* e componentes de DRM (*digital rights management*):

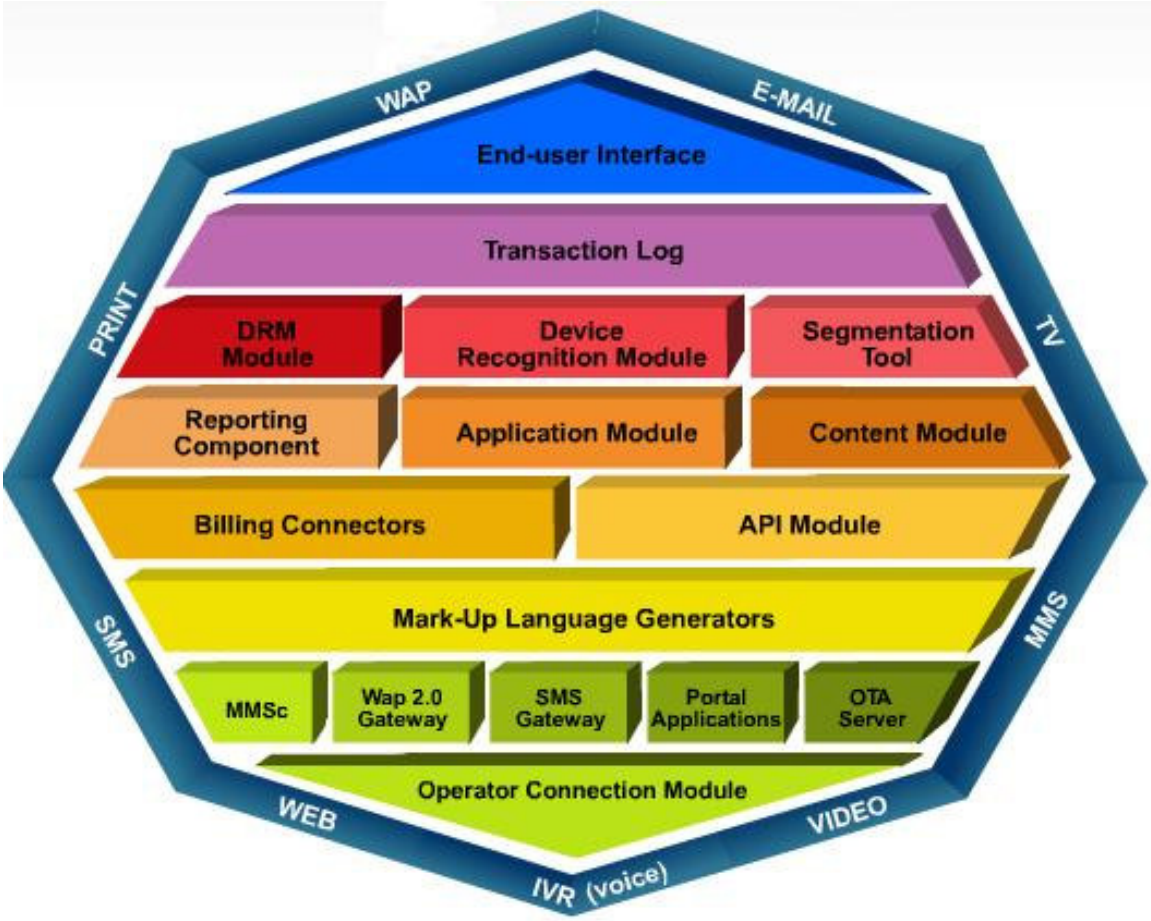


Figura 4 : Particularidades da plataforma

2.2.Conteúdos Disponibilizados

A TIMw.e. disponibiliza vários tipos de produtos, alguns de produção própria, outros pelos quais apenas somos responsáveis pela distribuição. A título de exemplo, temos os seguintes:

- Jogos (JAVA)
- Toques (polifónicos, reais e monofónicos)
- *Wallpapers*
- *Screensavers*
- Conteúdos via *streaming*
- Vídeos
- Lotarias
- Jogos (via SMS)
- Conteúdos de texto
- *E-Cards*

Estes produtos são a fonte de receitas da empresa, dado que são todos dirigidos para o uso por intermédio de dispositivos móveis.

2.3. Serviços Existentes

Entre as várias tarefas discriminadas para o 1º trimestre do estágio, existiu a possibilidade de receber formação sobre diversos produtos utilizados no negócio da empresa, entre os quais destaco:

- *Call centers*:

Interfaces para resolução de problemas dos clientes. Nestas interfaces existem, entre outras, as possibilidades de bloquear ou desbloquear um determinado cliente;

- *Wapsites*:

Sites orientados para dispositivos móveis, nomeadamente telemóveis, os quais possuem diferentes mecânicas relativamente aos websites, estão divididos em duas secções distintas: acesso a conteúdos directos (o link disponibilizado aponta para um conteúdo concreto), acesso a um conjunto de conteúdos (o *link* disponibilizado aponta para um menu de categorias de conteúdos, os quais são acedidos por intermédio de navegação *WAP*);

- Clubes de Subscrição:

Grande parte do negócio foca-se neste modelo de distribuição de conteúdos, através da disponibilização de toques (polifónicos/reais), *wallpapers*, jogos, mensagens de texto, entre outros. O cliente, quando se subscreve, aceita pagar por períodos de tempo fixo (semanalmente ou mensalmente) uma quantia fixa e identificada aquando do início da subscrição, permitindo que este aceda a um número também pré-definido de conteúdos;

- *One-Shot*:

Distribuição de conteúdos sem necessidade de subscrição num clube, estes são publicitados via *Webspot*, *website*, anúncios de TV ou anúncios *offline*. Após fazer o *download* do conteúdo e de ser cobrado, o cliente não recebe mais nenhum SMS relativo a qualquer campanha existente.

- *Webspots*:

Sites orientados a clubes ou conteúdos específicos, os quais podem definir o tipo de acesso, quer por subscrição, quer por *download* directo. O acesso aos clubes ou conteúdos por parte de um cliente pode ser feito após inserção de número de telefone, operador (opcional), e uma *password* (disponibilizada pelo nosso serviço);

- *Websites*:

Grande montra de conteúdos que possibilita ao cliente navegar por várias páginas e escolher os conteúdos e/ou clubes que deseja subscrever ou aceder. Como é o caso do *website* disponibilizado pela marca da empresa (NATTA);

- *Chats*:

Mecanismo que providencia a conversação entre diversos utilizadores. Estes possuem um *nickname* e podem aceder a *chatrooms* quer via *WAP*, quer via *SMS*.

Estes produtos representam o *business core* da empresa e a sua introdução foi gradual. Esta expansão deveu-se, tanto ao aumento da procura por parte de clientes, como ao contínuo crescimento da diversidade de conteúdos.

Para além da formação acima descrita, tive a possibilidade de começar a organizar a informação necessária à realização do meu projecto. Tanto em termos de ferramentas que irei utilizar (para simulação de acções que um cliente faria), como em termos de linguagens de programação, como ainda, em termos de objectivos de uso do próprio *Webspot*.

2.4. Interacção Entre Serviços

Algumas destas tecnologias atrás referenciadas interagem entre si. *Webspots* e *websites* têm como objectivos publicitar e orientar o cliente para serviços de subscrição e/ou serviços de busca de conteúdos sem fidelização do cliente (*one-shots*). Os *wapsites* podem também funcionar como meios publicitários de conteúdos, mas têm como objectivos principais apenas o *download* de conteúdos a que o cliente tem direito, quer por intermédio de uma subscrição, quer por intermédio de um *one-shot*. Os *call centers* funcionam como ferramenta de informação sobre estatísticas dos clubes de subscrição, funcionando como uma ferramenta de *reporting*.

2.5.A Minha Integração

A integração na empresa correu da melhor maneira possível, quer a nível interpessoal, onde todas as pessoas se mostraram disponíveis para me mostrarem os "cantos à casa", quer a nível tecnológico, onde verifiquei que as tecnologias usadas eram, na sua grande maioria, tecnologias com as quais eu já estava familiarizado, pelo meu percurso académico.

Tive a possibilidade de interagir, não só com todos os elementos de tecnologias de informação (IT), mas também com elementos das mais variadas secções, nomeadamente, de *marketing*, contabilidade, entre outras. Existe assim uma grande aprendizagem paralela de elementos de negócio, com os quais eu não estava anteriormente familiarizado. Esta interacção é fomentada, quer pelo espaço, dado que trabalhamos todos em espaços contíguos, quer pelo conceito do negócio em si, estando os *developers* (grupo do qual faço parte) com acesso privilegiado à secção de *marketing*, responsável pela criação do negócio, e podendo trocar ideias e encontrar as soluções que melhor responderão a vertentes mais especificas do serviço.

A nível do IT, a divisão feita por grupos de trabalho permite-nos um enriquecimento e troca de ideias mais objectiva, no que toca aos projectos em que estamos inseridos. Os grupos encontram-se inseridos num espaço comum, optimizando assim a comunicação entre os diferentes elementos. Estes são formados por equipas de três ou quatro *developers*, um *tester* (apenas presente em equipas cuja implementação de serviços para o país necessita de uma qualidade acima da média) e liderados por um gestor de projecto. Este tipo de equipa permite uma cooperação com a qual já estou familiarizado, pois os grupos na faculdade tinham um funcionamento semelhante, com excepção do papel do gestor de projecto, normalmente nunca assumido por nenhum aluno em especial, mas dividido por todos os elementos do grupo.

O grupo de trabalho no qual estou inserido é composto por quatro *developers* e um gestor de projecto, sendo que todos interagem entre si quando necessário, na da realização de tarefas, cooperando para a realização e consequente bom funcionamento do projecto dos Estados Unidos da América.

A comunicação entre os diversos departamentos ou grupos de um departamento é feita via *skype* para uma comunicação mais informal, ou por *salesforce*, onde são definidas e

geridas as tarefas a executar. Estas tarefas têm uma "vida" delimitada pela sua criação e a sua conclusão, passando as fases intermédias por vários grupos ou etapas de execução, desde a análise (análise dos recursos necessários, bem como a definição do tempo de desenvolvimento necessários para a conclusão da tarefa), desenvolvimento (fase na qual o *developer* está a executar a tarefa), testes em ambiente de desenvolvimento (feitos pelo *developer*, *testers* e/ou pelo responsável de *marketing* que criou a tarefa), testes em ambientes de produção (feito pelos *testers* e/ou pelo responsável de *marketing* que criou a tarefa), passagem de ambiente de desenvolvimento para ambiente de produção, entre outras. As tarefas são por norma criadas pelo departamento de *marketing*, sendo que as fases posteriores à sua criação são identificadas pelo departamento do IT.

Senti-me assim com todos os meios necessários para poder realizar o meu projecto de uma forma rigorosa, englobando todas as fases necessárias de análise, planeamento, especificação, desenho, desenvolvimento e testes de software para todas as ferramentas e aplicações de entretenimento necessárias, que possam surgir durante o meu estágio na empresa, entre elas, as já identificadas de início: *wapsites*; clubes de subscrição; *one-shots*; *Webspots*; *call-centers*; *chats*; *websites*.

3. Planeamento do Projecto

O desenvolvimento de *Webspots* é utilizado como principal mecanismo de divulgação de conteúdos e consequente angariação de clientes. Estes interagem com diversas tecnologias utilizadas pela empresa, nomeadamente publicitação de serviços de *one-shots* e clubes de subscrição. Estes últimos existem em grande quantidade, o que origina por sua vez, um maior número de *Webspots*, dado que, a título de exemplo, um clube de subscrição poderá possuir vários *Webspots* que orientam o cliente para os vários conteúdos desse clube.

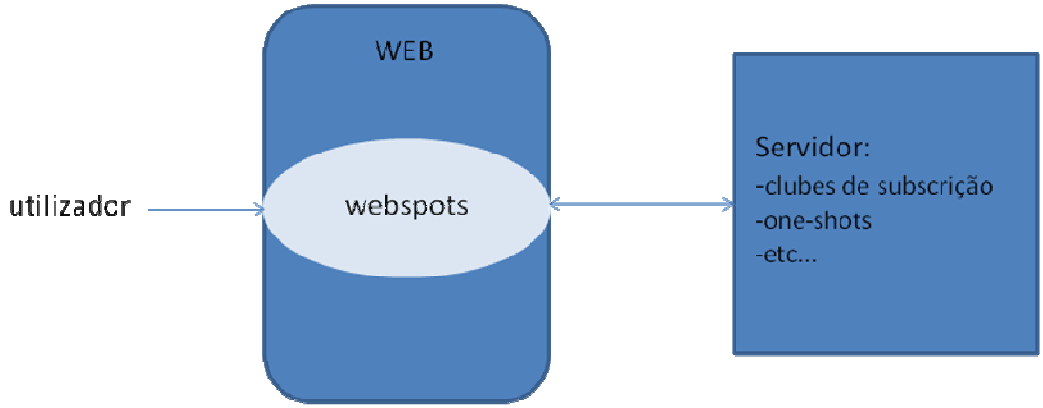


Figura 5: Interacção de um *Webspot*

A realização do projecto segue uma metodologia semelhante à utilizada nos projectos já desenvolvidos ao longo do meu percurso académico, envolvendo levantamento e análise de requisitos, desenho, seguido de implementação, testes e posterior manutenção e melhoramentos. É um projecto que engloba vários tipos de linguagens (JAVA, SQL, JSP, entre outras) e várias ferramentas de desenvolvimento, entre as quais poderei destacar o Eclipse e o SQL Developer. O projecto foi desenvolvido em várias etapas, na sua grande maioria, etapas com diferentes linguagens, estas depreenderam alguma aprendizagem (por exemplo JSP), outras, apenas o seu aperfeiçoamento (por exemplo PL/SQL). No final de cada etapa foi feita uma avaliação de tudo o que até aí foi feito, e o que se iria realizar de seguida. Para isso, a cooperação estreita com os responsáveis do *marketing* da região foi fundamental, de forma a evitar, desenvolvimentos desnecessários ou incorrectos.

3.1.Modelo de desenvolvimento do projecto

O desenvolvimento da *framework* irá ser feito em cascata, uma vez que será feito em várias etapas sequenciais, compreendendo:

- Levantamento de requisitos;
- Desenvolvimento a nível de base de dados;
- Desenvolvimento do projecto em JAVA e JSP;
- Desenvolvimento de algumas imagens e animações;
- Testes em ambiente de desenvolvimento;
- Correcção de possíveis *bugs*;
- Integração em ambiente de produção;
- Testes em ambiente de produção;
- *Release* do produto (utilização do produto para desenvolvimento de *Webspots*).

Todas estas etapas necessitam de validação do gestor de projecto e do comercial responsável pelo país. Estes mantêm-se assim a par de todo o desenvolvimento tendo em conta os requisitos pré-definidos.

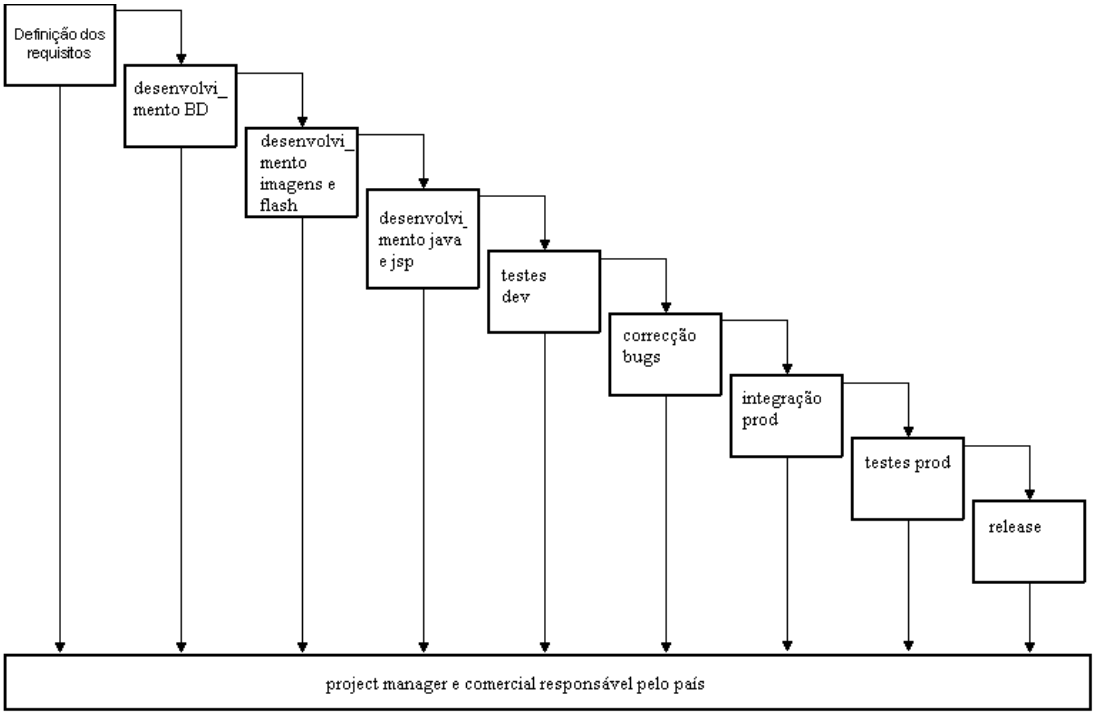


Figura 6: Diagrama do desenvolvimento da *Framework*

3.2. Calendarização e Plano de trabalho

O projecto a desenvolver envolve várias tarefas, a saber:

- Formação a nível das normas e métodos de trabalho da empresa:

  Leitura de documentação e formação sobre o tipo de negócio da empresa, bem como dos seus métodos de trabalho.

- Formação a nível dos serviços utilizados:

  Leitura de documentação e experimentação dos seguintes serviços:

  - *Webspots*
  - Clubes de subscrição
  - *One-shots*
  - *Wapsites*
  - *Websites*
  - *Call centers*
  - *Chats*

- Formação a nível das ferramentas utilizadas:

  Formação sobre as ferramentas utilizadas para desenvolvimento das tecnologias:

  - Simuladores de envio de mensagens (jetspeed)
  - Simuladores de telemóveis (Openwave)
  - Ambiente de desenvolvimento para diversas linguagens de programação (Eclipse)
  - Ambiente de desenvolvimento de SQL (SQL Developer)
  - Cliente FTP (WinSCP)

- Participações em projectos reais:

  Participação em projectos cuja finalidade é a distribuição de serviços ao público, fomentando o negócio da empresa

  - Criação do projecto dos Estados Unidos da América
  - Desenvolvimento da *framework* de desenvolvimento de *Webspot*s
  - Criação de serviços para os EUA assim como para outros países

❖ Apresentação de parceiros internos à empresa

Apresentação dos diferentes departamentos da empresa.

- Criação das imagens-tipo de *Webspot*s e de animações, em cooperação com o departamento de produção
- Interacção com o departamento de marketing para discussão dos serviços requisitados

❖ Manutenção e reformulação de projectos efectuados

- Descoberta e correcção de *bugs* identificados em projectos já em funcionamento
- Adicionar *features* novas a projectos já existentes

❖ Elaboração do relatório preliminar do estágio

❖ Elaboração do relatório final de estágio

Apenas algumas destas tarefas se confinam a certos trimestres do estágio, a maioria delas realiza-se ao longo do estágio.

Foi-me permitido o acesso a toda a documentação requisitada, de forma a optimizar o tempo de formação que nos foi disponibilizado. Essa formação foi feita por activos da empresa, os quais nos apresentaram de uma forma bastante acessível os métodos e parte das tecnologias e ferramentas utilizadas. A aprendizagem complementou-se *in loco,* através do nosso trabalho nos projectos em que estamos envolvidos.

Em seguida estão demonstradas as tarefas identificadas no modelo de desenvolvimento do projecto, bem como as tarefas existentes na calendarização e plano de trabalho, sob a forma de um mapa de Gantt, :

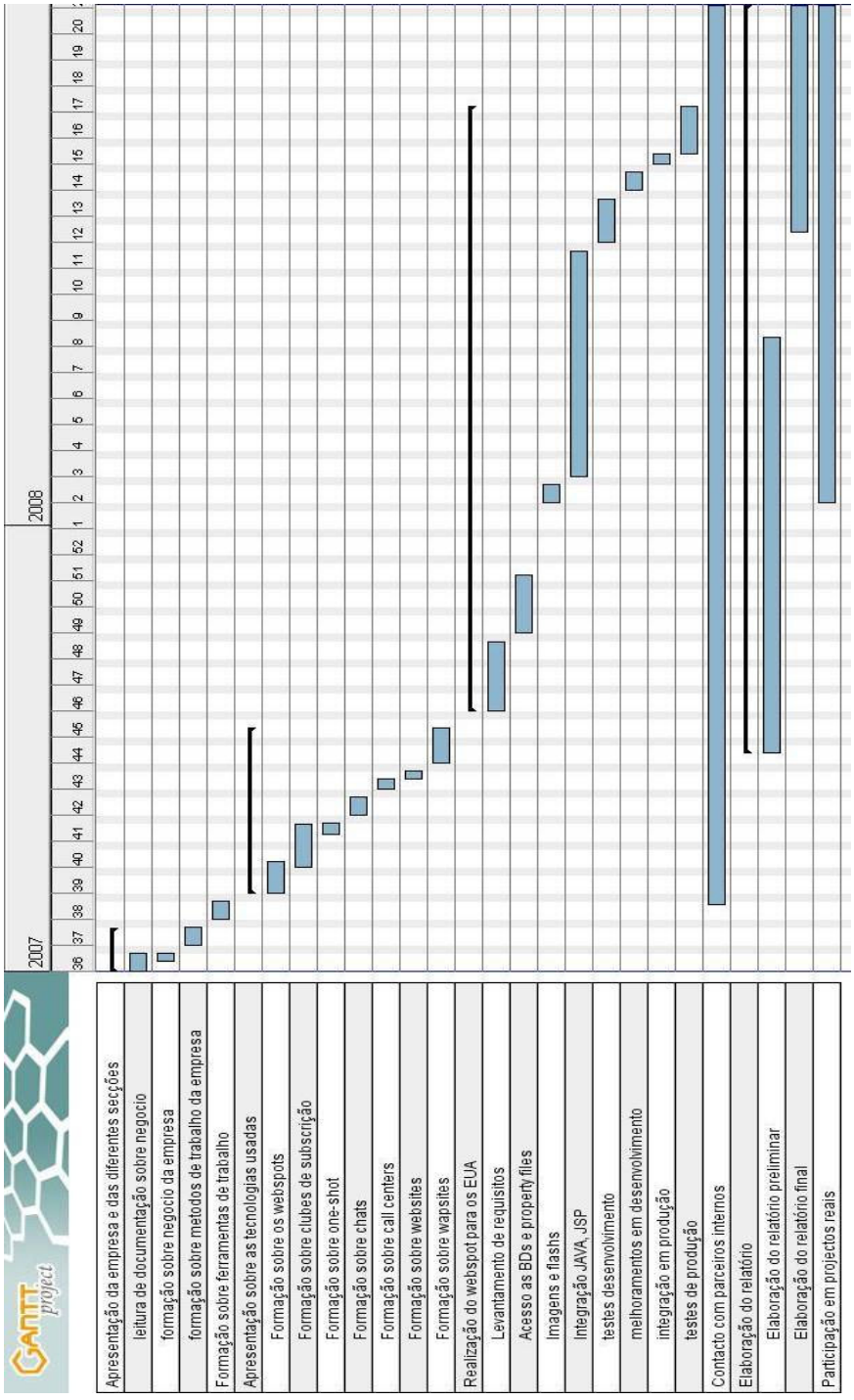


Figura 7: Mapa de Gant das tarefas

4. *Webspot*s

Neste capítulo são abordados todos os conceitos relativos aos Webspots, desde a sua arquitectura, casos-de-uso, funcionalidades, entre outros.

4.1. Conceito de Webspot

Webspot é um conceito criado de forma a responder mais objectivamente a vertentes do negócio relacionadas com publicidade via *web*. Permite uma orientação mais directa para o cliente, fazendo com que este apenas aceda ao que pretende, eliminando passos intermédios ou conteúdos menos interessantes no momento. Distribuído virtualmente via *web*, é apresentado por vezes como anúncio, sendo que a sua escolha para apresentação é decidida consoante o conteúdo que publicita.

4.2. Versões

Existem duas versões de *Webspot*s:

- A primeira (e mais utilizada) define como navegação um máximo de 4 páginas, sendo que todas as operações feitas (necessárias ao país em questão) têm obrigatoriamente de ser feitas nessas 4 páginas, sendo o fluxo entre elas contínuo (sequencial).

- A segunda versão ainda em estado de início de desenvolvimento, feito pela equipa responsável pela plataforma da empresa, irá permitir uma navegação por várias páginas, sem ordem concreta, muito à semelhança de um *website* e poderá ter páginas próprias para as funções a desenvolver. Esta versão ainda está numa fase inicial, pois a sua utilização assemelha-se muito a um *website* e ainda se estão a finalizar algumas características que poderá ter e que não sejam disponibilizadas num *website*.

4.3. Arquitecturas

As seguintes arquitecturas interagem entre si em alguns componentes, sendo que esta interacção é criada devido as especificidades dos *Webspots*.

4.3.1. Arquitectura dos *Webspots*

Webspots interagem com diversos serviços utilizados pela empresa, os principais são os clubes de subscrição e os serviços de one-shot. Os *Webspots* têm como objectivo reencaminhar e publicitar os clientes para estes serviços. As suas configurações são definidas em ficheiros de propriedades, podendo a sua interface ser manipulada com facilidade.

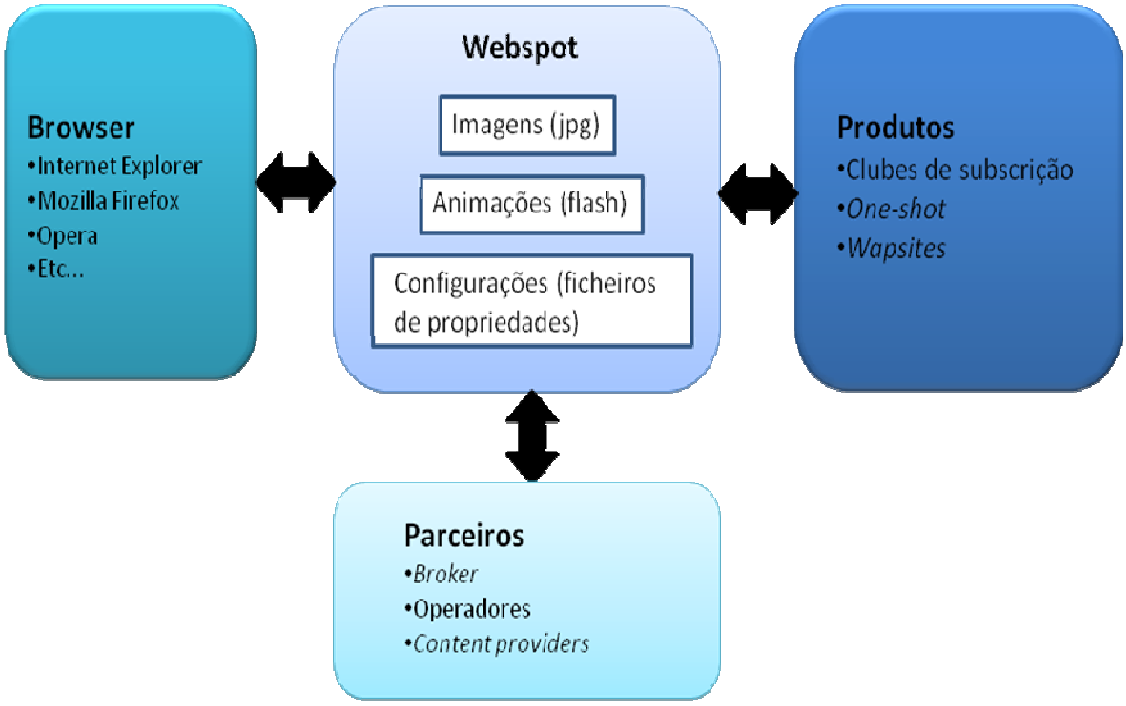


Figura 8: Arquitectura de *Webspot*

4.3.2. Arquitectura da Framework

A *framework* é responsável pela definição de várias operações que deverão ser suportadas por todos os *Webspots* de um país, engloba operações gerais a outras *frameworks*, bem como operações específicas para o país. Uma das principais funções desta *framework* consiste na delegação de tarefas noutras plataformas, assim como suportar as operações de configuração dos *Webspots*. Estas configurações podem ser definidas tanto em ficheiros de propriedades como na base de dados.

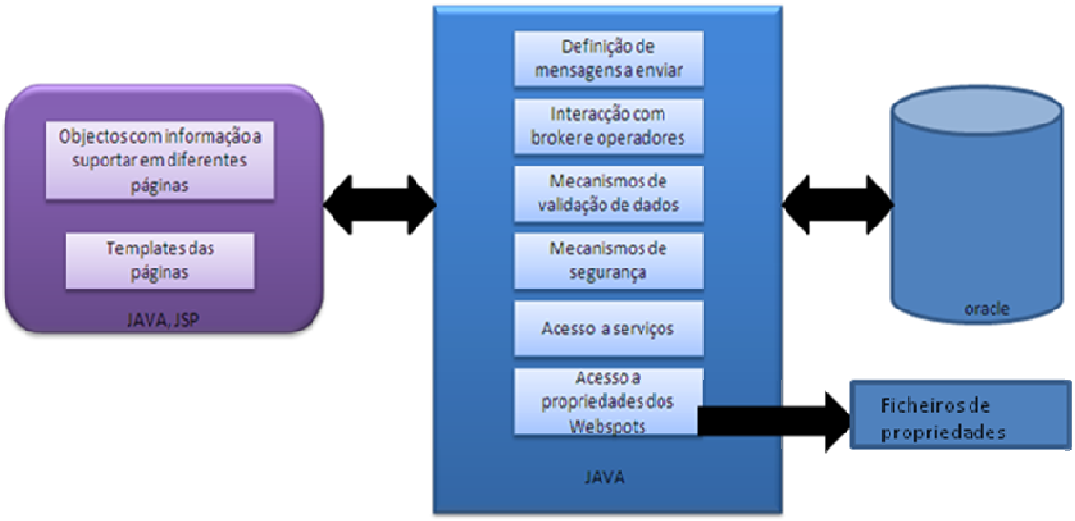


Figura 9: Arquitectura da *framework*

4.3.3. Arquitectura do Plataforma de Envio, Recepção e Tratamento de Mensagens

Os *Webspots* assentam em plataformas que fornecem recepção e envio de mensagens e conteúdos via *web*, para dispositivos móveis (figura 10). Estas mensagens têm origem nos telemóveis dos utilizadores, passando pelas operadoras e/ou *brokers*, até chegarem por via de uma *xconn* à nossa *gateway* de recepção/envio de mensagens. De seguida as mensagens são processadas numa *framework* própria para desenvolvimento de aplicações de *SMS*. Depois de processadas essas mensagens são enviadas para as aplicações correspondentes,

quer sejam elas clubes de subscrição, *one-shots*, *chats*, etc. Esta arquitectura é responsável pelo envio das mensagens criadas pela *framework* para cada *Webspot*.

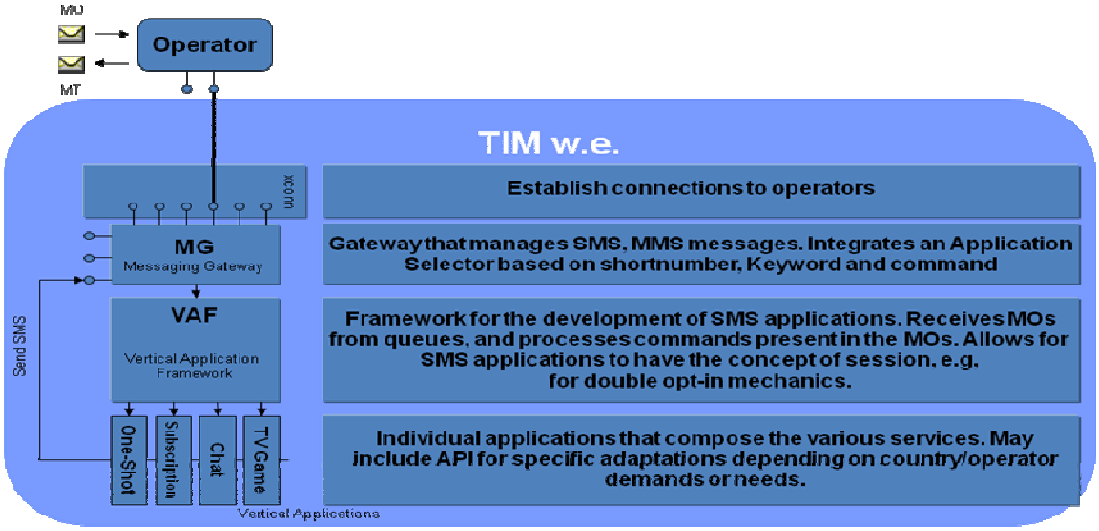


Figura 10: Arquitectura do sistema

4.4.Mecanismos e linguagens utilizados

Um *Webspot* engloba a aglomeração de vários tipos de mecanismos e linguagens, sendo que parte da sua informação é guardada em ficheiros de propriedades, enquanto que a restante é acedida numa base de dados. A sua implementação é feita através das linguagens: JAVA, SQL, PL/SQL, JSP. São também usados como interface gráfica de um *Webspot*, imagens em formato JPEG e animações, sendo que, por norma apenas a primeira página de um *Webspot* contem imagens em animações.

4.5.Funcionalidades

Os Webspots permitem uma grande variedade de vertentes de acesso e níveis de autenticação, com o objectivo possibilitar uma configuração de acordo com os clubes ou conteúdos que o cliente quer aceder.

4.5.1. Vertentes

Os *Webspots* funcionam de acordo com três vertentes distintas, definidas consoante os conteúdos a aceder:

- Acesso directo a conteúdos, também denominada de *one-sho*t – estas apontam directamente ao que o utilizador pretende, quer sejam jogos, vídeos, músicas, etc;
- Clubes de conteúdos – a sua subscrição via *Webspot* significa o acesso por intermédio da *web* a clubes que fornecem, vários tipos de conteúdos, sendo este clubes renovados diariamente, semanalmente ou mensalmente;
- Reencaminhamento do cliente para um *wapsite* de subscrição de um clube – promoção de clubes de subscrição que apenas conseguem ser acedidos via *wap*.

4.5.2. Níveis de Autenticação

Existem diversos tipos de acesso a conteúdos permitidos por um *Webspot*. Estes são definidos consoante o tipo de serviço que o utilizador está a aceder.

Após a introdução do número do telemóvel, do operador (opcional), da aceitação dos termos do serviço, o utilizador irá receber a sua *password* no seu telemóvel. Após a sua introdução no *Webspot*, são permitidos ao utilizador diferentes tipos de acesso:

- Acesso via *one-shot*:

o utilizador receberá o conteúdo no seu telemóvel sem qualquer outra informação adicional.

- Acesso via subscrição sem *double-optin*

o utilizador subscreve um clube, tendo por primeira mensagem recebida um conteúdo do mesmo.

- Acesso via subscrição com *double-optin*

o utilizador recebe uma mensagem para confirmação da subscrição, e só após a sua confirmação, este subscreve o clube, tendo por primeira mensagem recebida um conteúdo do mesmo.

- Acesso via subscrição com *triple-optin* (menos comum)

o utilizador recebe uma mensagem para confirmação da subscrição. Após a sua confirmação, este recebe uma terceira mensagem, normalmente relativa a permissão de acesso a conteúdos apenas disponibilizados para certas idades, e apenas após o compromisso por parte do cliente (de que a sua faixa etária se encontra dentro dos limites legais), este subscreve o clube, tendo por primeira mensagem recebida um conteúdo do mesmo.

4.5.3. Relacão entre Vertentes e Níveis de Autenticação

A vertente de acesso directo a conteúdos está directamente relacionada com o acesso one-shot, dado que este acesso não subentende nenhum tipo de subscrição, sendo apenas para compra a avulso de conteúdos. Os restantes níveis de autenticação poderão funcionar em ambas as vertentes de subscrição (directamente no clube, ou por intermédio dum *wapsite*), sendo que a diferença verificada se encontra na mensagem enviada ao cliente após a subscrição: no caso do clube de subscrição será uma mensagem com um conteúdo do clube; no caso do *wapsite*, será uma mensagem com o link de acesso ao *wapsite*.

4.6.Actores e descrição funcional

Actor	Descrição
Utilizador	O utilizador é pessoa que usufrui da aplicação O utilizador tem de introduzir o número de telefone, o operador (opcional) e aceitar os termos e condições O utilizador poderá, mais tarde receber um SMS com a sua *password* para confirmação da acção (subscrição ou recepção de conteúdo) O utilizador poderá ter de confirmar ainda que possui idade suficiente para receber o conteúdo

Tabela 1: Descrição do actor no processo de utilização de um *Webspot*

Área Funcional	Descrição
Introdução dos dados do utilizador	Introdução dos dados do utilizador no *Webspot*
Validação do utilizador	Recepção da *password* no telemóvel Introdução da *password* no *Webspot*
Verificação da idade	Validação da idade do utilizador tendo em conta a faixa etária definida para o serviço/conteúdo no *Webspot*
Confirmação	Mensagem no *Webspot*, que confirma a subscrição/acesso a conteúdo

Tabela 2: Descrição da tabela funcional geral de um *Webspot*

4.7.Casos de uso

Em seguida são apresentados os casos de uso para as operações gerais suportadas pelos *Webspot*s. Duas delas apresentam-se como obrigatórias: Introdução de dados por parte do utilizador e validação do mesmo, de forma a permitir o acesso aos clubes e conteúdos publicitados no *Webspot*. Existe uma terceira operação que apenas é aplicada em alguns países, a verificação da idade do utilizador, de forma a precaver o público do acesso não controlado a todos os conteúdos disponibilizados.

4.7.1. Subscrição num clube de conteúdos

Nome	Susbcrição num clube de conteúdos
Descrição	Utilizador acede ao *Webspot*
Actor principal	Utilizador
Caminho principal	1. Utilizador introduz o número de telemóvel, operador 2. Introdução do operador do cliente (esta opção pode ser feita pelo sistema (1), ou pelo cliente(2))

	3. Utilizador aceita os termos e condições 4. Utilizador introduz idade no *Webspot* (opcional) 5. Utilizador recebe (no telemóvel) *password* a introduzir na 2ª página 6. Utilizador introduz a *password* 7. Utilizador verifica a mensagem de sucesso disponibilizada no *Webspot* 8. Utilizador recebe mensagem de confirmação de subscrição num clube e/ou o conteúdo seleccionado
Caminho secundário	1.1 Utilizador introduz número de telemóvel incorrecto, ou não selecciona operador, ou não aceita os termos e condições 1.2 Utilizador recebe mensagem a indicar que os campos obrigatórios não foram preenchidos/foram mal preenchidos 1.3 Utilizador não acede ao *Webspot* 2.1 Utilizador introduz idade inferior ao limite estabelecido 2.2 Utilizador recebe mensagem a indicar que o conteúdo não é indicado para a sua idade 2.3Utilizador não acede ao *Webspot* 3.1 Utilizador introduz *password* errada 3.2 Utilizador recebe mensagem a indicar que a *password* é errada 3.3 Utilizador não acede ao *Webspot*

Tabela 3: Caso de uso geral de subscrição num clube de conteúdos

4.7.2. Acesso One-Shot a conteúdos

Nome	Acesso One-Shot a conteúdos
Descrição	Utilizador acede ao *Webspot*
Actor principal	Utilizador
Caminho principal	1Utilizador introduz o número de telemóvel, operador 9. Introdução do operador do cliente (esta opção pode ser feita pelo sistema (1), ou pelo cliente(2)) 10. Utilizador aceita os termos e condições 11. Utilizador introduz idade no *Webspot* (opcional) 12. Utilizador recebe (no telemóvel) *password* a introduzir na 2ª página 13. Utilizador introduz a *password* 14. Utilizador verifica a mensagem de sucesso disponibilizada no *Webspot* 15. Utilizador recebe mensagem com o conteúdo publicitado
Caminho secundário	1.3 Utilizador introduz número de telemóvel incorrecto, ou não selecciona operador, ou não aceita os termos e condições 1.4 Utilizador recebe mensagem a indicar que os campos obrigatórios não foram preenchidos/foram mal preenchidos 1.3 Utilizador não acede ao *Webspot* 2.3 Utilizador introduz idade inferior ao limite estabelecido 2.4 Utilizador recebe mensagem a indicar que o conteúdo não é indicado para a sua idade 2.3Utilizador não acede ao *Webspot*

	3.1 Utilizador introduz *password* errada 3.2 Utilizador recebe mensagem a indicar que a *password* é errada 3.3 Utilizador não acede ao *Webspot*

Tabela 4: Caso de uso geral acesso a conteúdos *one-shot*

4.8. Diagrama de Actividades

De seguida estão demonstrados os diagramas de actividades referentes aos casos de uso realizados numa sequência de páginas do *Webspot* (diferenciando apenas no alvo da acção desempenhada pelo *Webspot* (susbcrição ou *one-shot*)). São as acções executadas pelos utilizadores e sistema, bem como a sua sequência e as respostas gráficas do sistema.

4.8.1. Página inicial do *Webspot* (introdução de dados)

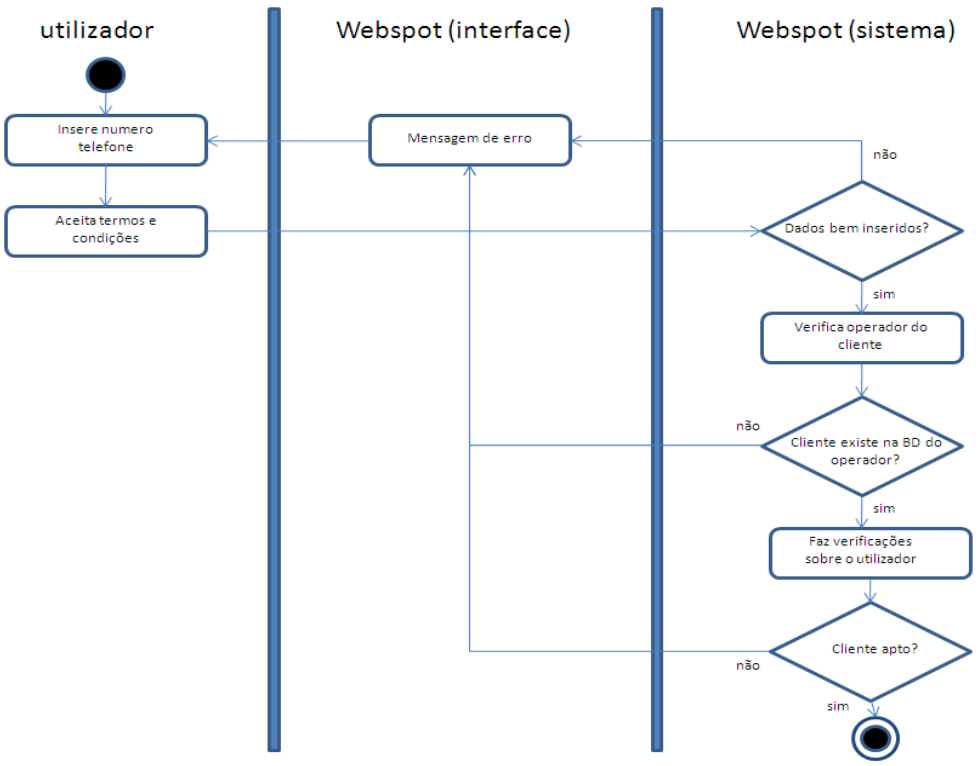


Figura 11: Diagrama de actividades da 1ª página do *Webspot*

4.8.2. Segunda página do *Webspot* (introdução da *password*)

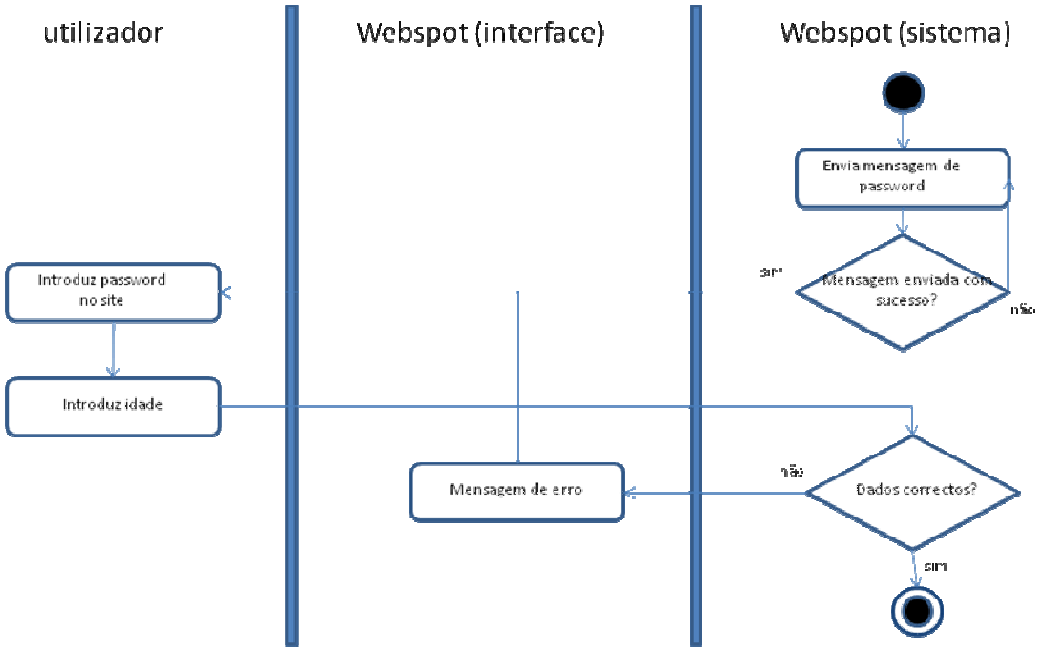


Figura 12: Diagrama de actividades da 2ª página do *Webspot*

4.8.3. Terceira página do *Webspot* (confirmação de subscrição)

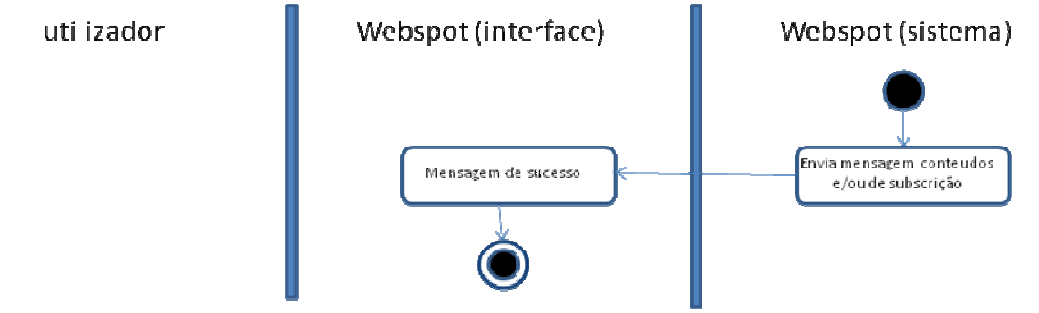


Figura 13: Diagrama de actividades da 3ª página do *Webspot*

4.9. Ciclo de criação de um *Webspot*

Em seguida demonstramos um exemplo do ciclo de desenvolvimento de um *Webspot*. Este ciclo é em tudo semelhante ao de outras tarefas (desenvolvimento de outros serviços) executadas, sendo que a sua semelhança advém dos estados, e não obrigatoriamente do que é feito em cada um, nomeadamente os estados de desenvolvimento e testes (para algumas tarefas apenas é obrigatória a aprovação do *tester* e não do comercial):

1. Submissão

Comerciais reúnem a informação necessária, em termos de grafismo, clubes associados, mecânicas, para o sucesso do *Webspot*, e submetem a tarefa ao gestor de projecto do país, dado que este é o responsável por toda a coordenação do trabalho feito para o país.

2. Análise

O gestor de projecto analisa o tempo de execução do *Webspot* e aloca um *developer* para o seu desenvolvimento. Caso falte alguma informação, por lapso do comercial, a tarefa (neste caso o *Webspot*) passa para o estado "*missing data*".

3. Desenvolvimento

Nesta fase, o *developer* está encarregue da identificação do *webspot*, a escolha do template (moldura JSP onde as imagens, HTML e animações serão inseridas), preenchimento dos ficheiros de propriedades com todas as informações do *Webspot*, inserção das imagens e animações. Caso a tarefa tenha de ser pausada, por exemplo, por alteração de prioridades, esta passará para o estado "*on hold*" (onde a tarefa aguarda que sejam terminadas acções que condicionam a continuação do seu desenvolvimento).

4. Testes em ambiente de desenvolvimento

Os testes efectuados são executados numa primeira instância pelo *developer*, de forma a corrigir todos os possíveis *bugs* existentes. De seguida, os testes estão ao encargo do *tester*

da equipa, ou dos comerciais. Este tipo de testes origina uma fase seguinte distinta, sendo que as tarefas testadas pelos primeiros, caso obtenham aprovação passam para o estado "*IT approved*", e as tarefas aprovadas pelos comerciais passam para o estado "*Stage approved*".

5. *Deployment*

Após aprovação, a implementação feita é passada para o CVS (controlo de versões), e é feito o pedido de passagem dessa informação para o ambiente de produção. Esta transição está a cargo da equipa de operações técnicas, responsável pelos *deployments* e pela manutenção do ambiente de produção.

6. Testes em ambiente de produção

Após a passagem para o ambiente de produção, são feitos testes já em ambiente real (ambiente aberto ao público) sendo estes testes feitos com a finalidade de verificar se o *deployment* ocorreu sem erros e se o serviço funciona correctamente. Aqui são feitos testes que não poderiam ser feitos em ambiente de desenvolvimento (por exemplo chamadas SOAP a alguns operadores, dado que apenas são possibilitadas ligações reais ao *broker* e operadores em ambiente de produção). Todos os testes nesta etapa são feitos recorrendo a números de telemóvel reais, e os custos inerentes também são os reais.

7. Finalização da tarefa

Caso a tarefa passe com sucesso nos testes de produção, o comercial é responsável pela sua conclusão, e pelo início da publicidade inerente.

Este ciclo de desenvolvimento é o utilizado em todas as tarefas realizadas na empresa, sendo que todas sem excepção terão de passar por estas etapas até serem lançadas para o ambiente de produção.

5. Projecto Realizado

Neste capítulo é apresentada a concepção e implementação da *framework* de suporte ao desenvolvimento de *Webspots* para os Estados Unidos da América, evidenciando as principais especificidades desta *framework* relativamente a outras já existentes.

A *framework* possui diversos mecanismos comuns a outras *frameworks*, como por exemplo:

- Possibilidade de oferecer primeira semana grátis

Outros mecanismos necessitaram de ser desenvolvidos, devido aos requisitos do *broker* com quem a empresa comunica nos Estados Unidos da América, como por exemplo:

- *webservices* que permitam extrair informação sobre o cliente. Informação essa que o *broker* detêm, ou que requisita aos operadores.

Estas particularidades motivam a criação de projectos de dimensões mais reduzidas, mas que apenas servem o propósito do país. Projectos esses que apoiam a estrutura da *framework*, como por exemplo projectos que reúnam todos os *webservices*, ou projectos que reúnam toda a informação necessária ao envio e recepção de mensagens. Esta divisão em pequenos projectos permite uma divisão das operações realizadas pela *framework*, evitando que diferentes operações se aglomerem num só projecto.

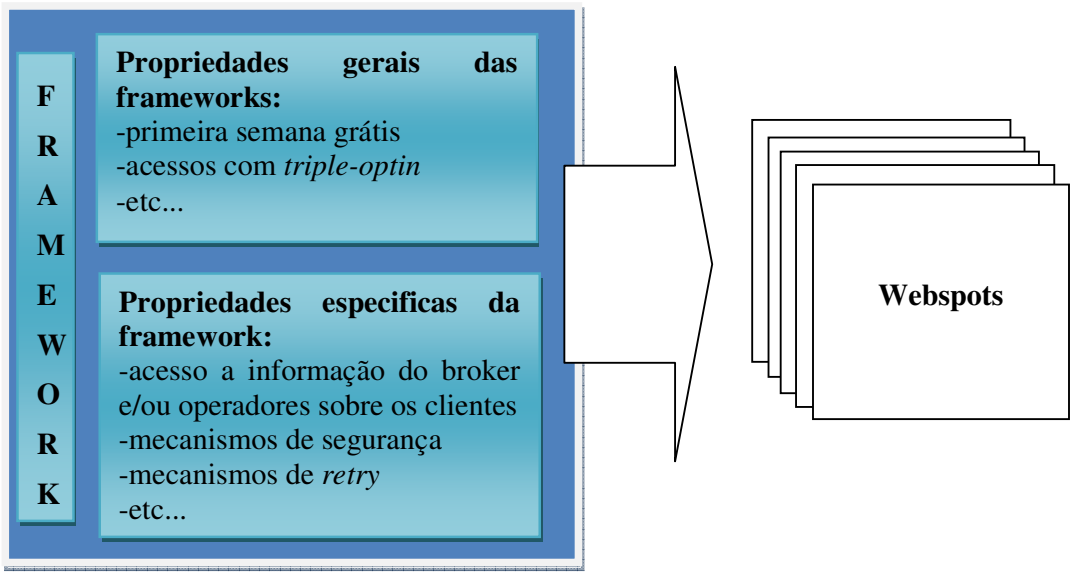


Figura 14: Diagrama de propriedades da *framework*

O projecto realizado foi feito segundo um modelo de desenvolvimento em cascata, sendo que as várias etapas se basearam no desenvolvimento e testes dos requisitos definidos no início do processo.

5.1.Análise de Requisitos

A análise de requisitos foi feita em conjunto com o gestor de projecto (responsável pela distribuição das tarefas em termos organizativos) e o comercial (conhecedor do mercado e responsável pelo negócio do país).

5.1.1. Requisitos da *Framework*

Após uma análise conjunta, e verificadas as necessidades do negócio (para ter sucesso naquele país) e os requisitos do *broker*, foram definidos os requisitos para a *framework* a ser concebida. Para além dos requisitos gerais estipulados por uma *framework*, os requisitos particulares são:

- Criação de *webservices* que acedem a um cliente localizado no *broker*, de forma a ser possível extrair informação sobre o cliente que está a aceder ao *Webspot* no momento:
    - Verifica se o cliente já excedeu o limite máximo de gastos num período de tempo – alguns operadores definem um limite de gastos para um mês, outros definem um limite de gastos para uma semana e por fim, alguns operadores não definem qualquer tipo de limite;
    - Verificação do operador do cliente – esta característica, permite que seja retirada dos *Webspot*s a *dropbox* que possui os diferentes operadores (para o cliente escolher), opção que não tem muito sucesso nos Estados Unidos da América;
    - Verifica se o cliente está *blacklisted* – os clientes podem ficar bloqueados no lado do *broker*, por várias razões, não tendo por isso acesso a navegação no *Webspot*;

- Criação de cliente de *Webservice* responsável pelo pedido de envio de *password*, feito apenas para um dos operadores que exigiu este mecanismo, sendo o operador responsável pela criação das *password*s.

- Mecanismo de envio de mensagens de lembrança com *retries* (novo reenvio caso a mensagem não alcance o seu destino) para introdução da *password* do cliente;

- *Broadcast* de mensagens informativas de um dado *Webspot*. Nomeadamente nos feriados dos Estados Unidos da América (*Halloween*, *St. Patrick's Day*, *Thanks Giving*, etc.);

- Mecanismos de contagens de *clicks* nos *Webspots*, para diferentes operadores e patrocinadores, para dados estatísticos ou comerciais;

- *AdSenses* [1], com informação sobre parceiros, de forma a promover alguns conteúdos que estes divulgam. Esta publicidade é recíproca, dado que também os nossos conteúdos são divulgados em *AdSenses* dos parceiros.

- Mecanismo de redireccionamento para outros *Webspots*, caso o cliente já esteja subscrito no clube que está ligado ao *Webspot* onde este está a navegar, ou apenas não possa subscrever esse clube, quer por limites de idades, quer por particularidades dos operadores;

- Envio seguro de *passwords* através de uma ligação segura SSL [2], para envio das *passwords* para alguns operadores;

- Mecanismos de segurança para os *Webspot*s, quer no envio de mensagens ao cliente, quer na possibilidade de criação de múltiplas subscrições, quer na possibilidade de ataques de força bruta ao *Webspot* (introdução de varias *password*s);

- Oferta de conteúdos promocionais disponibilizados por *content providers*;

- Oferta de um número variável de mensagens grátis ao cliente, consoante os clubes a que se subscreve;

Estes requisitos foram extraídos de diversas reuniões que efectuei em conjunto com a comercial responsável pelo país e o meu gestor de projecto. Nessas reuniões, analisámos documentos providenciados pelo *broker* e pelos operadores e estatísticas obtidas pela comercial, informação que nos permitiu ver tendências do público-alvo e particularidades técnicas exigidas pelos operadores e pelo *broker*. Nos Estados Unidos nota-se uma grande necessidade de simplificação do *layout* nos *Webspots*, o que obriga a que seja feito um trabalho mais complexo sem que o cliente se aperceba disso. É um público muito objectivo e que não gosta muito de demasiados passos intermédios. Exemplo disso é a preferência pela não existência de *dropboxes* com os operadores e pelas *boxes* de introdução dos números de telemóveis divididas, para facilitar a introdução dos indicativos, e restantes números.

5.2.Acesso e Configuração da Base de Dados e Ficheiros de Propriedades

Para a criação dos Webspots, é necessário a *framework* obter informação com diferentes finalidades. A informação de configuração da *framework* (classes responsáveis por vários tipos de processamento) foi guardada na base de dados, a informação relativa a configuração dos *Webspots* foi guardada em ficheiros de propriedades.

5.2.1. Base de Dados

Nesta etapa do trabalho, foram criadas as tabelas específicas de informação da *framework* na base de dados. Os acessos foram feitos na linguagem JAVA, e construídos ao mesmo tempo que o projecto da framework era inicializado. Foram criadas as querys necessárias para inserção, remoção e edição da informação contida na base de dados, de modo a serem despoletadas, em diferentes etapas da navegação do cliente no *Webspot*, consoante a

evolução da navegação. São feitas verificações nas tabelas que contêm informação sobre os clientes, informação sobre as subscrições, informação sobre as operadoras, e informação sobre as classes que irão executar certas partes do código (verificação de saldo, de operador, classe de envio de *password*, etc...).

5.2.2. Ficheiros de Propriedades

Os ficheiros de propriedades [3] são os grandes responsáveis pelo dinamismo dos *Webspot*s. Nestes, a informação sobre cada Webspot é definida, originando blocos de informação, sendo que cada bloco corresponde a informação de apenas um *Webspot*, como por exemplo números de páginas, tipo de informação presente na página, tipo de publicidade presente na página, caminhos para as imagens e animações utilizados, entre outros.

As propriedades são lidas por *pattern matching*, sendo que têm prioridade as propriedades de um Webspot sobre as propriedades gerais do país. Na figura seguinte está representado o exemplo de leitura de uma propriedade:

```
String googleAdClient = props.getProperty("spot.googleAdClient."
        + spotId, props.getProperty("spot.googleAdClient." + country,
        "blank"));
```

Figura 15: Exemplo de leitura de uma propriedade

As propriedades podem ser definidas de forma geral para todo o país ou de forma particular para cada *Webspot*, permitindo que este tenha uma definição diferente da existente para o país, sem influenciar os outros *Webspots*.

É possível definir nos ficheiros de propriedades, praticamente toda a informação necessária à configuração dos *Webspot*s. Esta vai ser introduzida no *template* (ficheiro JSP que irá construir a página *web*), e preencher os campos necessários à construção da página. As configurações das propriedades permitem que os templates possam ser muito dinâmicos, exemplo disto são o tipo de letra utilizada, e a cor dos textos. Nesta óptica de dinamismo podemos definir uma vasta gama de propriedades da página no ficheiro de propriedades. De seguida estão apresentadas as principais:

- Existência ou não da *tickbox* (na 1ª ou 2ª página)
- Texto da *tickbox*
- Existência ou não da *combobox* dos operadores
- Indicativos dos operadores existentes na *combobox*
- *Links* externos
- Cor de fundo da página
- Controlo do msisdn introduzido (adicionar-lhe indicativo do país)
- Definição de existência de *timer*
- Definição de existência de mecanismos de controlo de *clicks*
- Informação sobre mecanismos de *retry*
- Informação sobre *googleids*
- Informação sobre *adsenses*
- Informação sobre mecanismos de redireccionamento para outros *Webspot*s do mesmo país

Estas características são definidas tendo em conta a particularidade de cada *Webspot* ou de uma forma geral, sendo definidas por país.

De seguida mostramos exemplos das propriedades que podem ser definidas:

1) Exemplo representativo do ficheiro de propriedades dos *Webspot*s dos Estados Unidos (definição do indicativo e medidas de segurança)

```
#--------------------------------
# USA Info
#--------------------------------

spot.country.1001=1

alreadyin.wp.check.enabled=true
alreadyin.wp.check.max=2

free.pass.sms=true
free.pass.sms.max=5

susbscription.active.enable=true
susbscription.active.max=5
```

Figura 16: Exemplo de propriedades d*e* segurança de um *Webspot*

Na figura 16, podemos verificar informação sobre o indicativo do país para o *Webspot* em questão, bem como algumas propriedades que vão ser verificadas durante a acção de certos tipos de mecanismos de defesa:

- Mensagens indicativas de que o cliente já está subscrito no clube correspondente do *Webspot* limitadas a 2 por dia

- Mensagens de *password* enviadas para o cliente limitadas a 5

- Subscrições activas pelo cliente durante um dia, limitadas a 5

2) Exemplo representativo do ficheiro de propriedades dos *Webspot*s dos Estados Unidos e tabela com informação correspondente a cada propriedade

```
# LIGHT

spot.name.1001=Light - Ringtones, Wallpapers and more
spot.image1.1001=images/usa/4_00022628_INT_light_02.jpg
spot.image2.1001=images/usa/3_00022628_CONF_light_02.jpg
spot.image3.1001=images/usa/00020967_CONG_light.jpg
spot.flash.1001=flash/usa/00022628_INT_light.swf
spot.passText.1001=Pswrd: ## Enter it now and save it for future visits to our
spot.club.1001=2673
spot.feature.1001=1
spot.webtext.subinfo.1001=*Get this content now as part of Club Natta!  By ent
spot.webtext.conf.1001=You will be receiving shortly a text message with your
spot.featsize.1001=1
spot.image1.dimhigh.1001=600
spot.image2.dimhigh.1001=600
spot.image3.dimhigh.1001=600
spot.flash.dimhigh.1001=600
```

Figura 17: Exemplo de configurações de um *Webspot*

Informação disponibilizada	**Propriedade correspondente**
Título da página	spot.name.(spot id)
Termos e condições	spot.webtext.subinfo.(spot id)
Mensagem de password	spot.passtext.(spot id)
Mensagem de confirmação de introdução dos dados do utilizador	spot.webtext.conf
Caminhos para as imagens e animações	Spot.image1. (spot id) - imagem da primeira página spot.flash.(spot id) – animação utilizada no *webspot*
Template utilizado	spot.feature.(spot id)
Clube ao qual o Webspot está associado	spot.club(spot id)

Tabela 5: Propriedades de configuração apresentadas na figura 17

5.3.Definição de Imagens e *Animações*

Nesta etapa, foram construídas algumas imagens utilizadas em todos os *Webspot*s deste país, bem como as imagens e animações do *Webspot* que foi criado paralelamente à construção da *framework*, de forma a ajudar a testar o seu funcionamento. Nesta fase, contei com a ajuda de elementos da área de produção, os quais me orientaram e me deram os conhecimentos necessários para desenvolver as imagens e animações de um *Webspot*, nomeadamente em termos de localização de imagens, as zonas mais "chamativas" do ecrã, complementaridade entre imagem e animações.

As animações criadas possibilitam ao Webspot apresentar certas amostras de conteúdos que são disponibilizados pelo clube ao qual o cliente está a subscrever, como por exemplo *previews* de *wallpapers*, de músicas, etc.

Figura 18: Exemplo de *Webspot* com *previews* de músicas

As imagens desenvolvidas para teste da *framework* encontram-se representadas de seguida:

Figura 19: Imagem da pagina inicial do *Webspot*

Na figura 19 podemos ver o conjunto de animações e imagens inseridas numa página de HTML.Aqui, toda a parte superior (acima da caixa de fundo preto) é constituída por animações, sendo que tudo o que se encontra abaixo é imagem. É sobre a imagem que é implementado o código de onde se extrai toda a informação necessária. No caso presente esta é a imagem que esta no ambiente de produção. Temos os campos de introdução do telefone da pessoa, e uma pequena *tickbox* de confirmação de aceitação dos termos e condições. Neste ambiente não possuímos *combobox* de escolha do operador, pois temos possibilidade de obtê-lo via pedido SOAP ao operador. Caso que não se verifica em ambiente de desenvolvimento, onde forçamos o aparecimento dessa *combobox* por intermédio de inserção de parâmetros extra no *url*.

Figura 20: Imagem da página de confirmação de *password*

Na figura 20 podemos ver a representação da página de confirmação. Aqui apenas é mostrada a informação sobre o número do cliente e é demonstrado o *timer* que irá funcionar como mecanismo de *retry*. Por exemplo, quando uma mensagem não chega ao cliente dentro do tempo definido, é enviada uma mensagem num período de tempo mais curto, e assim sucessivamente. Em termos de informação visual o timer apenas serve para "agarrar" o cliente ao *Webspot*. A disposição das animações e do JPEG nesta página é igual à disposição encontrada na 1ª página.

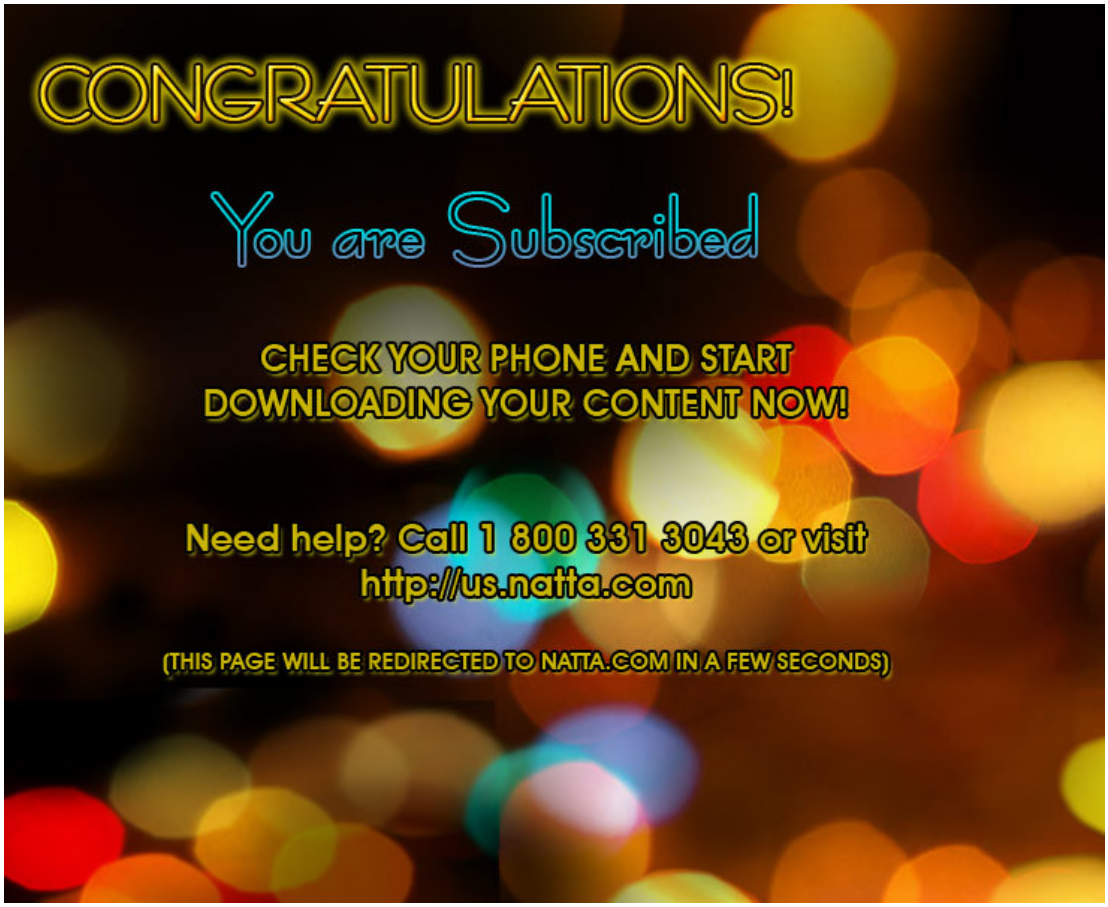


Figura 21: Imagem da página final do *Webspot*

Por fim, temos a página de confirmação, apenas construída por uma imagem. Esta contém informação necessária para o cliente e imposta pelo *broker*.

Esta sequência de páginas e disposição das imagens, animações e HTML corresponde apenas a um dos muitos *templates* definidos em JSP.

5.4. Desenvolvimento Java e JSP

Esta fase englobou a implementação de todos os requisitos resultantes da análise feita com a comercial responsável e o gestor de projecto, bem como os requisitos gerais verificados em todas as *frameworks*. Embora já implementados, estes terão de ser redefinidos e adaptados à realidade do país, sendo que a mecânica irá ser semelhante às já existentes.

A implementação ocorreu em duas fases:

- a construção de templates utilizando HTML e JSP que numa primeira instância possuíam definições constantes(com o objectivo da definição dos vários *templates*), aprovados após a sua finalização pelo comercial e o gestor de projecto;
- o desenvolvimento efectuado a nível do JAVA abarca todos os mecanismos necessários ao funcionamento dos *Webspots*, do mais simples (por exemplo, verificação do número de telemóvel introduzido), até ao mais complexo (por exemplo, mecanismo de *retry* de envio de mensagens de *password* consoante a passagem do tempo). Após a conclusão da *framework* poderão ainda ser feitas modificações resultantes de novos acordos com novos operadores, ou de mudanças de critério dos operadores e do *broker* com quem a empresa já estabeleceu negócio.

5.4.1. Definição de *templates* em HTML/JSP

Foram definidos os templates a serem usados na construção dos *Webspots*. Estes foram definidos recorrendo a linguagens tais como o HTML e o JSP [4] e tiveram como objectivo a definição de páginas de acordo com as melhores opções do negócio, assim decididas pelo comercial responsável. Numa primeira instância, estes templates possuíam valores estáticos (cor de fundo, tipos de letra, mensagens no ecrã, etc.). Após a conclusão do projecto JAVA, estes valores irão ser dinâmicos e definidos nos ficheiros de propriedades. Foram desenvolvidos vários *templates*. Em seguida encontram-se exemplificados alguns deles, já com imagens, animações e texto definidos:

Figura 22: Exemplo de template para a primeira página de um *Webspot*

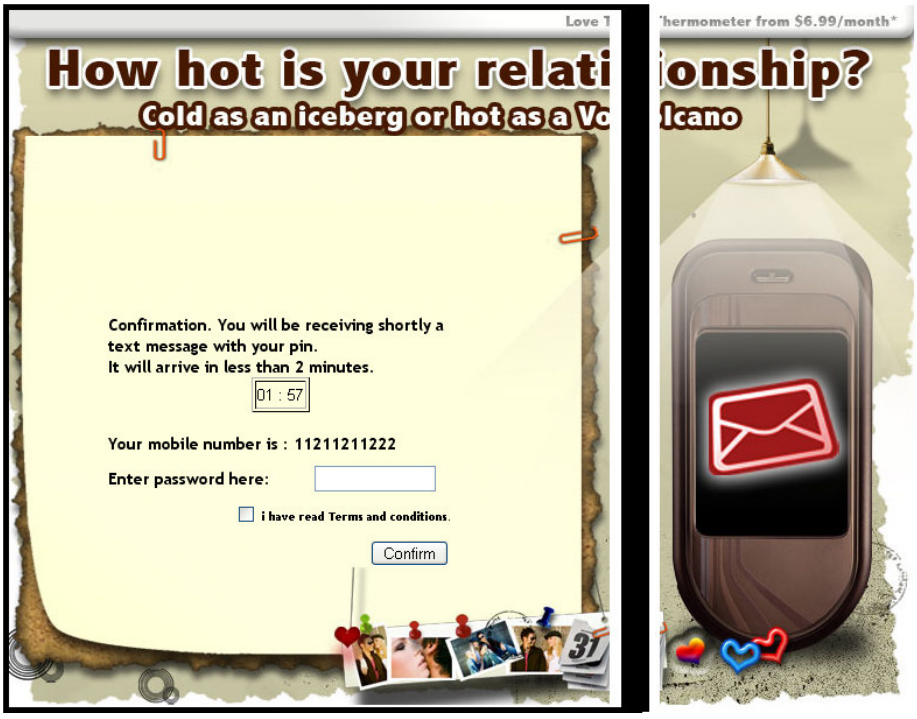


Figura 23: Exemplo de template para a segunda página de um *Webspot*

Figura 24: Exemplo de template para a primeira página de um *Webspot*

Figura 25: Exemplo de template para a segunda página de um *Webspot*

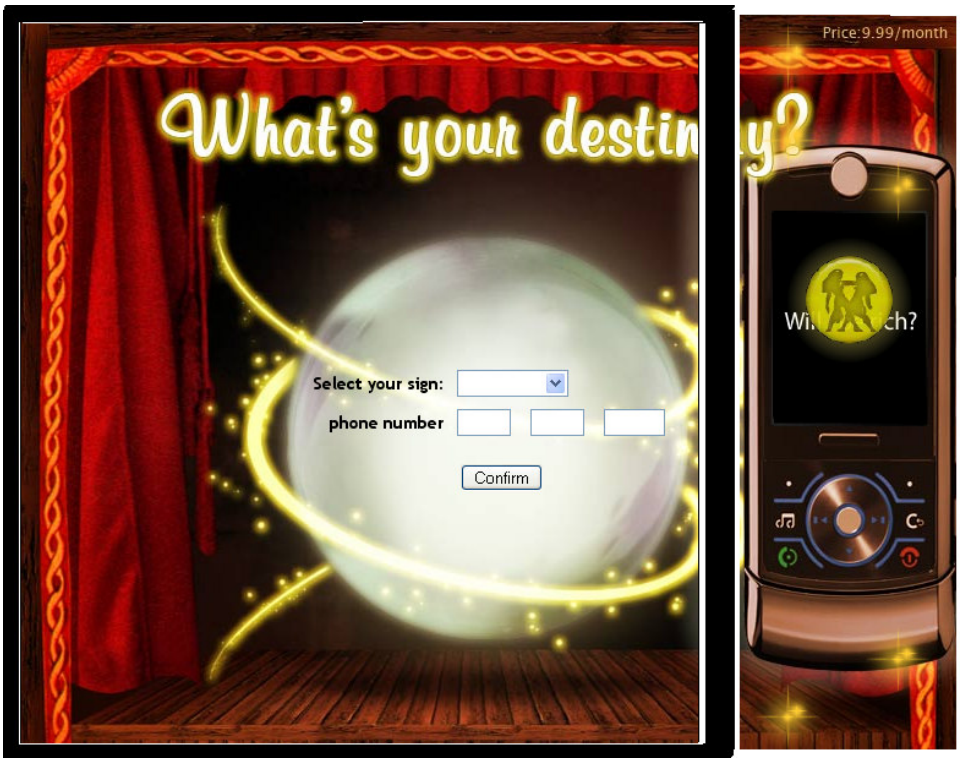


Figura 26: Exemplo de template para a primeira página de um *Webspot*

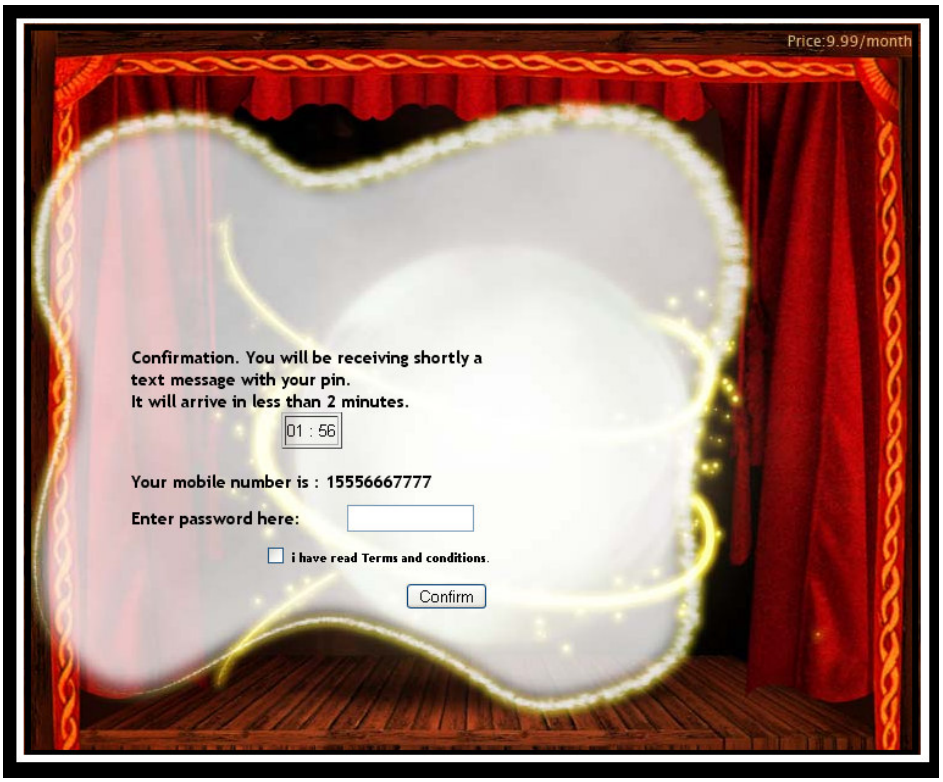


Figura 27: Exemplo de template para a segunda página de um *Webspot*

Nos *templates* atrás identificados (figuras 22 a 27), podemos ver as diferentes disposições das imagens e das animações. Para facilitar a sua identificação nas figuras, as imagens encontram-se destacadas por uma moldura preta. Esta disposição é pedida pelo comercial aquando do pedido de construção de um *Webspot*, e com o sentido de optimizar o seu desenvolvimento. Os templates foram definidos aquando do desenvolvimento da *framework*, de forma a orientar os *Webspot*s para um padrão estabelecido de formatos, e evitar constantes mudanças nos *templates*.

Para além das mudanças em termos de disposição no grafismo, estão também presentes algumas diferenças em termos de campos a preencher, nomeadamente as duas caixas para preenchimento dos nomes para acesso, neste caso, ao clube termómetro (figura 22), bem como da *combobox* que disponibiliza uma lista de signos dos quais o cliente irá escolher o seu (figura 26). Estes campos são definidos tendo em conta o clube para o qual o *Webspot* irá direccionar o utilizador. Podem existir campos para preenchimento de nomes, escolha de signos, escolha de conteúdos, entre outros. Todas estas modificações são definidas de duas maneiras, quer por propriedades definidas nos ficheiros de propriedades (por exemplo, identificação de todos os signos existentes, num clube horóscopo), quer num template próprio para esse clube (existência de uma *combobox* para escolha do signo). Várias propriedades poderão aparecer dispostas no *Webspot* caso haja uma indicação para tal, no *url*. A título de exemplo, temos a *combobox* para escolha do operador do cliente (usada em ambiente de desenvolvimento, visto os números de testes utilizados não serem necessariamente números atribuídos a telemóveis, e como tal, a chamada feita pelo webservice não obteria resposta). Podemos ver na figura 24 a existência dessa *combobox*, a qual foi activada adicionando um parâmetro adicional ao url do *Webspot*:

- *Webspot* com *combobox* para escolha do operador:

  http://www.vibramovel.com/ws/acp?sp=10025&combo=969

- *Webspot* sem *combobox* para escolha do operador:

  http://www.vibramovel.com/ws/acp?sp=10025

A definição e verificação de parâmetros nos ficheiros JSP são funções definidas num ficheiro TLD (*Tag Library Descriptor*) [5], o qual após ser inicializado no início do template, fornece as funções desejadas e por ele permitidas (definições, verificações, comparações entre parametros, etc...). De seguida temos exemplos da construção de um desses ficheiros, a forma como são chamados no ficheiro JSP e por fim, a forma como são utilizadas as suas variáveis na construção da página:

- Exemplo de construção do ficheiro TLD:

```
<tlibversion>1.0</tlibversion>
<jspversion>1.1</jspversion>
<shortname>bean</shortname>
<uri>http://jakarta.apache.org/struts/tags-bean</uri>
<tag>
     <name>define</name>
     <tagclass>org.apache.struts.taglib.bean.DefineTag</tagc
lass>
     <teiclass>org.apache.struts.taglib.bean.DefineTei</teic
lass>
     <bodycontent>JSP</bodycontent>
     <attribute>
          <name>name</name>
          <required>false</required>
          <rtexprvalue>true</rtexprvalue>
     </attribute>
     <attribute>
          <name>id</name>
          <required>true</required>
          <rtexprvalue>false</rtexprvalue>
     </attribute>
     <attribute>
```

```
            <name>property</name>
            <required>false</required>
            <rtexprvalue>true</rtexprvalue>
        </attribute>
        <attribute>
            <name>type</name>
            <required>false</required>
            <rtexprvalue>true</rtexprvalue>
        </attribute>
        <attribute>
            <name>value</name>
            <required>false</required>
            <rtexprvalue>true</rtexprvalue>
        </attribute>
(...)
</tag>
```

AS definições realizadas permitem que estes atributos, quando acedidos, construam objectos das classes acima indicadas (ex: <tagclass>org.apache.struts.taglib.bean. DefineTag </tagclass>, define uma variável de script baseada nos valores do *bean* q a constrói) possibilitando o seu uso na construção da página HTML por parte dos ficheiros JSP.

- Exemplo de inicialização do ficheiro TLD no ficheiro de construção da página (JSP):

```
<%@ taglib uri="/WEB-INF/struts-html.tld"
prefix="html"%>
<%@ taglib uri="/WEB-INF/struts-logic.tld"
prefix="logic"%>
<%@ taglib uri="/WEB-INF/struts-bean.tld"
prefix="bean"%>
<%@ page contentType="text/html; charset=utf-8"%>
```

- Exemplo de referência a um atributo do objecto criado (no caso, o objecto bean):

```
    <bean:define    id="bgColourScr"    property="bgColour"
name="beSpot"/>
    <body bgcolor=<%=bgColour%>>
```

Neste exemplo faz-se a ligação entre o objecto beSpot e o ficheiro JSP que está no momento a construir a página, podendo aceder a propriedade bgColour, definida no objecto e definindo-a como uma variável de script chamada bgColourScr.

5.4.2. Implementação JAVA

A componente JAVA[6][7] desenvolvida nesta *framework* englobou várias classes (anexo 1) e métodos essenciais para o funcionamento dos websptos, assim como servlets, webservices e algumas classes particulares, para os Estados Unidos da América.

De seguida estão identificadas algumas das classes básicas para navegação, criadas nesta *framework*:

- BeSpot – Classe que funciona como um *bean*, responsável pela acumulação de informação contida nos ficheiros de propriedades, ou de informação obtida por operações feitas durante a navegação do *Webspot* (número de telémovel, operador, etc...). Esta classe é estendida por três outras, que para além das características herdadas, possuem também:

    - BeLogin – características necessárias ao *login* do utilizador;
    - BeValid – características necessárias a validação do utilizador;
    - BeConf – características necessárias a confirmação de sucesso da operação feita pelo utilizador no *Webspot*.

- AcSpot – Classe responsável pela construção do *Webspot*. Toda a aparência do *Webspot* é feita nesta classe. É feita também a verificação dos campos a preencher pelo utilizador. Aqui são executadas algumas das chamadas ao *webservice* com o objectivo

de executar algumas verificações do cliente (se o cliente está *blacklisted*; qual o operador do cliente) no *broker*. O *bean* utilizado nesta classe é o definido pela classe BeLogin.

- AcSpotSendPassword – Classe responsável pelo tratamento da informação introduzida pelo utilizador, envio de mensagens de *password* e de algumas das operações feitas com os operadores/*broker*. De seguida estão demonstradas as funções realizadas por esta classe, quer em métodos nela contidos, quer em acessos a métodos de outras classes:
  - Pesquisas na base de dados para verificar se o cliente já está subscrito no clube representado no *Webspot*, e redireccioná-lo para uma página de outro *Webspot* que publicite um clube, no qual, o utilizador ainda não está subscrito;
  - Envio de mensagens de *password* para o cliente via SSL;
  - Envio limitado de mensagens de *password* ou de "*already in*" para o cliente por dia. Este mecanismo de segurança evita que *hackers* ou *BOTs* congestionem a rede com mensagens "*free*". Estas têm sempre um custo para a empresa e, caso estejam a ocupar a rede, colocam em espera, por grandes períodos de tempo mensagens que têm como destino utilizadores reais;
  - Mecanismo de contagem de *passwords* erradas introduzidas por um cliente por dia. Evita que um *hacker* ou um *BOT* encontre uma *password* que não lhe pertence, recorrendo a força bruta (tentado todas as combinações existentes);
  - Envio de mensagens publicitárias de outros clubes para o cliente;
  - Acesso a uma classe responsável pelo mecanismo de reenvio de mensagem, quando o *timer* localizado na página atingir o valor 0. Este reenvio é feito tendo em conta o número de tentativas já efectuado e o tempo entre cada uma;
  - Chamada SOAP utilizando a API providenciada pelo operador, para este criar e enviar a *password* ao cliente. Esta *password* também nos é enviada (sendo posteriormente actualizada na nossa base de dados) para serem permitidos acessos dos utilizados ao nosso *website*.
  - Geração de *passwords* para os utilizadores

  O *bean* utilizado nesta classe é o definido pela classe BeValid.

- AcSpotSubmitPassword – Classe responsável pela criação da subscrição do utilizador num clube, bem como de envio das mensagens de início de subscrição e alguns conteúdos. É na 3ª página de um *Webspot* (construída por esta classe) que são contabilizados os cliques feitos pelos utilizadores nos *Webspot*s. Existem diversas empresas que contabilizam estes cliques, de forma a monitorizar o acesso dos utilizador e orientar a sua publicidade consoante o tipo de acessos. A sua publicidade é feita nos *adsenses*, onde os *links* lá contidos poderão referenciar conteúdos dessas mesmas empresas. Esta contabilização de cliques é feita de várias maneiras, podendo ir de um simples conjunto de parâmetros indicados no url da página gerada (é feita uma verificação a existência destes parâmetros, caso existam, é actualizado o valor de cliques feito até ao momento, na nossa base de dados), até uma chamada HTTP feita aquando da construção da página, a um servidor da empresa (contabilização de cliques feita do lado de outra empresa). O *bean* utilizado nesta classe é definido pela classe BeConf.

Outras classes foram definidas para comportar as classes anteriormente definidas, nomeadamente em termos de objectos usados, sessões, dados cifrados, classes de envio de mensagens grátis, entre outras.

A interacção entre as classes é feita por intermédio de acções, estas determinam os caminhos a seguir, consoante o resultado das operações efectuadas na classe. Para cada acção determinada existe um reencaminhamento, este é usado para enviar o cliente para diferentes páginas consoante o resultado das operações efectuadas na página onde este se encontra no momento. As acções identificam a classe JAVA responsável pela construção da página, a sua identificação no URL e todos os reencaminhamentos que esta classe suporta.

De seguida estão representados alguns exemplos do uso destas acções, bem como o significado dos reencaminhamentos:

- Exemplo de redireccionamento de uma acção:

```
<form-beans>
  <form-bean name="beConf"
   type="com.mindergy.webspot.beans.BeConf" />
</form-beans>
<action path="/acSpotSubmitPassword"
        type="com.mindergy.webspot.actions.AcSpotSubmitPassw
   ord"
        name=" beConf " unknown="false" scope="session">
  <forward name="success" path="/end.do" />
  <forward name="badPassword" path="/enterPassword.do" />
  <forward name="error" path="/acSpot.do" />
  <forward name="home" path="/spot.do" />
/>
</action>

<action path="/spot" forward="/spots/spot.jsp"
   unknown="false" />
 <action path="/enterPassword"
   forward="/spots/enterPassword.jsp" unknown="false" />
 <action path="/end" forward="/spots/end.jsp"
   unknown="false" />
```

Neste exemplo estão identificadas as acções de redireccionamento para vários cenários, quando o cliente esta a submeter a password. A titulo de exemplo temos o erro na introdução da password, este reencaminhará o cliente novamente para a página de introdução da password (funcionará neste caso como um *refresh*)

5.4.3. Interacção entre Ficheiros de Propriedades, JSP e JAVA

De seguida estão exemplificadas diferentes fases de implementação de um pedaço de informação relativa aos *adsenses*:

1. A informação de configuração do adsense é introduzida num ficheiro de propriedades, de forma a poder ser alterada com relativa flexibilidade:

```
spot.googleAdSenseFirstPage.1001=true
spot.googleAdClient.1001=pub-2325747866421521
spot.googleAdWidth.1001=120
spot.googleAdHeight.1001=600
spot.googleAdFormat.1001=120x600 as
spot.googleAdType.1001=text_image
spot.googleAdChannel.1001=7553486668
spot.googleAdBorder.1001=000000
spot.googleAdBg.1001=F0F0F0
spot.googleAdLink.1001=0000FF
spot.googleAdText.1001=000000
spot.googleAdUrl.1001=008000
spot.googleAdSrc.1001=http://pagead2.googlesyndication.com/pagead/show_ads.js
```

Figura 28: Exemplo da definição de *AdSenses* num ficheiro de propriedades

Na figura 28 podemos ver os diferentes parâmetros necessários para a construção de um *adsense*, incluindo, para além da cor do texto, tamanho e largura do *adsense*, também o ficheiro *JavaScript* que irá accionar.

2. Esta informação é, de seguida, lida nas classes JAVA (alguns exemplos em seguida):

```
//AdSense Properties
String googleAdSenseFirstPage = props.getProperty("spot.googleAdSenseFirstPage."
        + spotId, props.getProperty("spot.googleAdSenseFirstPage." + country,
        "false"));

bs.setGoogleAdSenseFirstPage(googleAdSenseFirstPage.equals("true"));
Log.info("AdSense-googleAdSenseFirstPage : " + googleAdSenseFirstPage);

String googleAdClient = props.getProperty("spot.googleAdClient."
        + spotId, props.getProperty("spot.googleAdClient." + country,
        "blank"));
bs.setGoogleAdClient(googleAdClient);
Log.info("AdSense-googleAdClient : " + googleAdClient);
```

Figura 29: Exemplo de leitura de características de um AdSense numa classe JAVA

Na figura 29 estão descritos alguns exemplos de leitura dessas propriedades, na sequência a mesma descrita anteriormente, propriedades definidas por *Webspot*, têm mais prioridade que propriedades definidas para um país.

3. Por fim o *adsense* é construído a nivel do JSP:

```
<td width="100%" align="right" valign="top">
    <iframe src ="/ws/ads.do" width="<%=googleWidth%>"
    height = "<%=googleHeight%>" scrolling="no"
```

Figura 30: Exemplo de leitura de um *AdSense* num ficheiro JSP

Na figura 30 podemos ver a leitura de algumas das propriedades, anteriormente definidas e actualizadas pelas classes que se encontra na componte JAVA, por parte do ficheiro JSP, de forma a criar código HTML da página.

5.5.Testes em Ambiente de Desenvolvimento e em Ambiente de Produção

Estes testes foram executados, após a implementação da *framework* e do *Webspot* que foi criado para o propósito de testes. Teve como objectivo descobrir *bugs*, quer na implementação da *framework*, quer em termos de mecanismos utilizados, foi executado em cooperação com o tester da equipa, e simulando todas as interacções em termos de *webservices* e números de telemóvel.

Os testes efectuados foram testes de "caixa – cinzenta", sendo que foram feitos com o objectivo de verificar se o *output* gerado era o esperado consoante o *input* inicial, e ao mesmo tempo verificar se a aplicação é robusta em termos internos, se não apresenta erros em zonas problemáticas da mesma.

6. Conclusão

A *framework* foi desenvolvida na sua totalidade e permitiu que a construção de *Webspots* se tornasse muito mais simples e rápida, necessitando apenas de configurações (a nivel dos ficheiros de propriedades) para o *Webspot* ser implementado.

O desenvolvimento web que foi executado, juntamente com a criação de tabelas em bases de dados e consequente consulta, inserção e remoção de informação das mesmas, criação de webservices, mecanismos de segurança e mecanismos de envio e recepção de mensagens, permitiram que a construção desta *framework* aglomera-se um grande conjunto de tecnologias.

Em termos técnicos, este trabalho proporcionou-me o desenvolvimento das minhas capacidades de programação, nomeadamente o aperfeiçoamento em temos de linguagem JAVA, SQL, PL/SQL, assim como a aprendizagem de novas linguagens de programação (como por exemplo JSP) e protocolos de envio e recepção de mensagens (SMPP e HTTP).

Após a finalização da *framework* foi-me possível ter uma visão global sobre o tipo de publicidade feito aos conteúdos disponibilizados e a forma como são feitas algumas das interacções com parceiros externos a empresa, no caso *brokers* e operadores. Tive oportunidade também de aprender quais os factores que levam a que certos tipos de conteúdos sejam preteridos em relação a outros, e quais as formas de negócio mais indicadas para os diversos países.

Em termos pessoais, permitiu-me ter uma noção do funcionamento duma empresa que opera neste tipo de negócio (resposta rápida ao cliente), bem como de tudo o que engloba um ambiente profissional (relações interpessoais, etc.)

Em termos de trabalho a desenvolver no futuro, estará a implementação de possíveis comunicações com novos operadores, bem como a expansão da plataforma para englobar particularidades de *brokers* e operadores canadianos (projecto que estará também ao cargo da mesma equipa, da qual eu faço parte).

7. Glossário

Adsense – anúncio disponibilizado na web;

Animações – ficheiro gerado pelo Adobe Flash (antigamente pelo Macromedia Flash) para animações multimédia;

Anúncios offline – anúncios feitos a conteúdos, nomeadamente em revistas;

API – conjunto de rotinas e padrões estabelecidos por um software para utilização de suas funcionalidades por programas aplicativos;

Bean – objecto que possui toda informação necessária às classes que o utilizam;

BOT – programa informático que replica automaticamente vários tipos de operações;

Broker – entidade intermediária entre a TIMwe e os operadores, responsável pela aglomeração de um ou mais operadores, permite a uniformização de acessos TIMwe <--> operadores;

Bugs – erros descobertos aquando do desenvolvimento ou teste de uma aplicação;

Combobox – caixa de selecção de diferentes conteúdos;

Content providers – empresas parceiras da TIMw.e. responsável pela disponibilização de certos conteúdos;

Conteúdo – objecto acedido pelo utilizador final, aquando de um acesso;

Deployment – passagem dos projectos do ambiente de desenvolvimento para o ambiente de produção;

Developer – programador;

Framework – estrutura de suporte definida em que outro projecto de software pode ser organizado e desenvolvido;

IT – information technology;

LA – número utilizado pela TIMw.e. para os serviços (por norma difere de pais para pais, podendo excepcionalmente, um pais, possuir mais que uma);

Pattern Matching – reconhecimento de padrões, através deste tipo de mecanismo, podem ser reconhecidos os diferentes tipos de dados que devem ser lidos;

Pesquisa directa – pesquisa feita por um conteúdo específico;

SSL (Secure Socket Layer) – protocolos criptográficos que provêem comunicação segura para transferência de dados;

SMPP (short message peer-to-peer) – protocolo de envio e recepção de SMS

SOAP – protocolo para troca de informações estruturadas em uma plataforma descentralizada e distribuída, utilizando tecnologias baseadas em XML;

Tester – pessoa responsável por assegurar (por intermédio de testes) a qualidade do software desenvolvido;

Tickbox – caixa de selecção de opção única (opção activa ou desactivada);

Utilizador – pessoa, possuidora de uma telemóvel, que tenciona obter conteúdos para o mesmo, disponibilizados pela empresa;

Xconn – ligação SMPP a um broker ou operador

8. Bibliografia

[1] Google Advertising Programs**:** http://www.google.com/intl/en/ads/

[2] Secure Socket Layer:

- http://en.wikipedia.org/wiki/Secure_Sockets_Layer
- http://java.sun.com/j2se/1.5.0/docs/api/

[3] Property files: http://commons.apache.org/configuration/howto_properties.html

[4] Java Server Pages: http://java.sun.com/products/jsp/

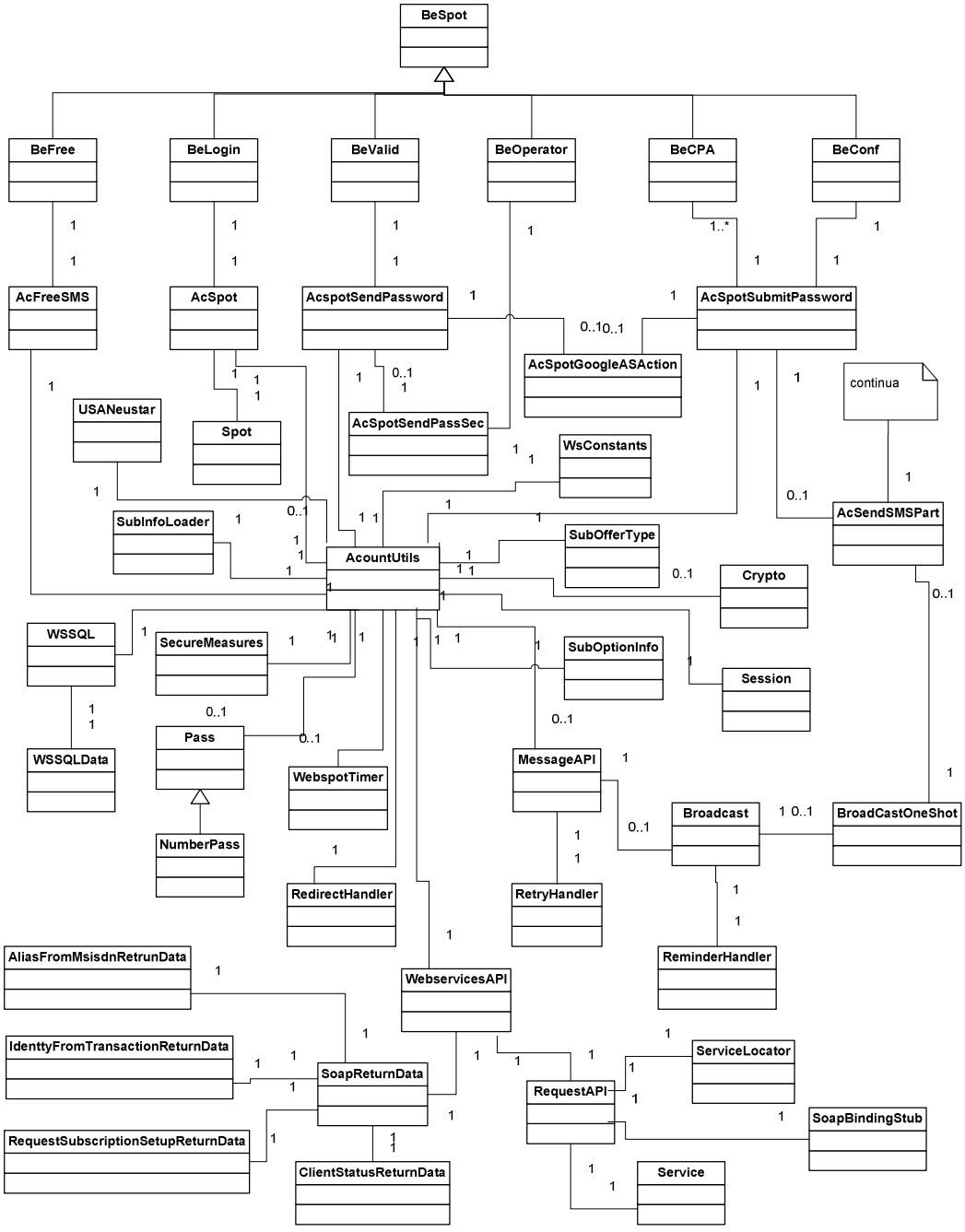
[5] Tag Library Description: http://java.sun.com/j2ee/1.4/docs/tutorial/doc/JSPTags6.html

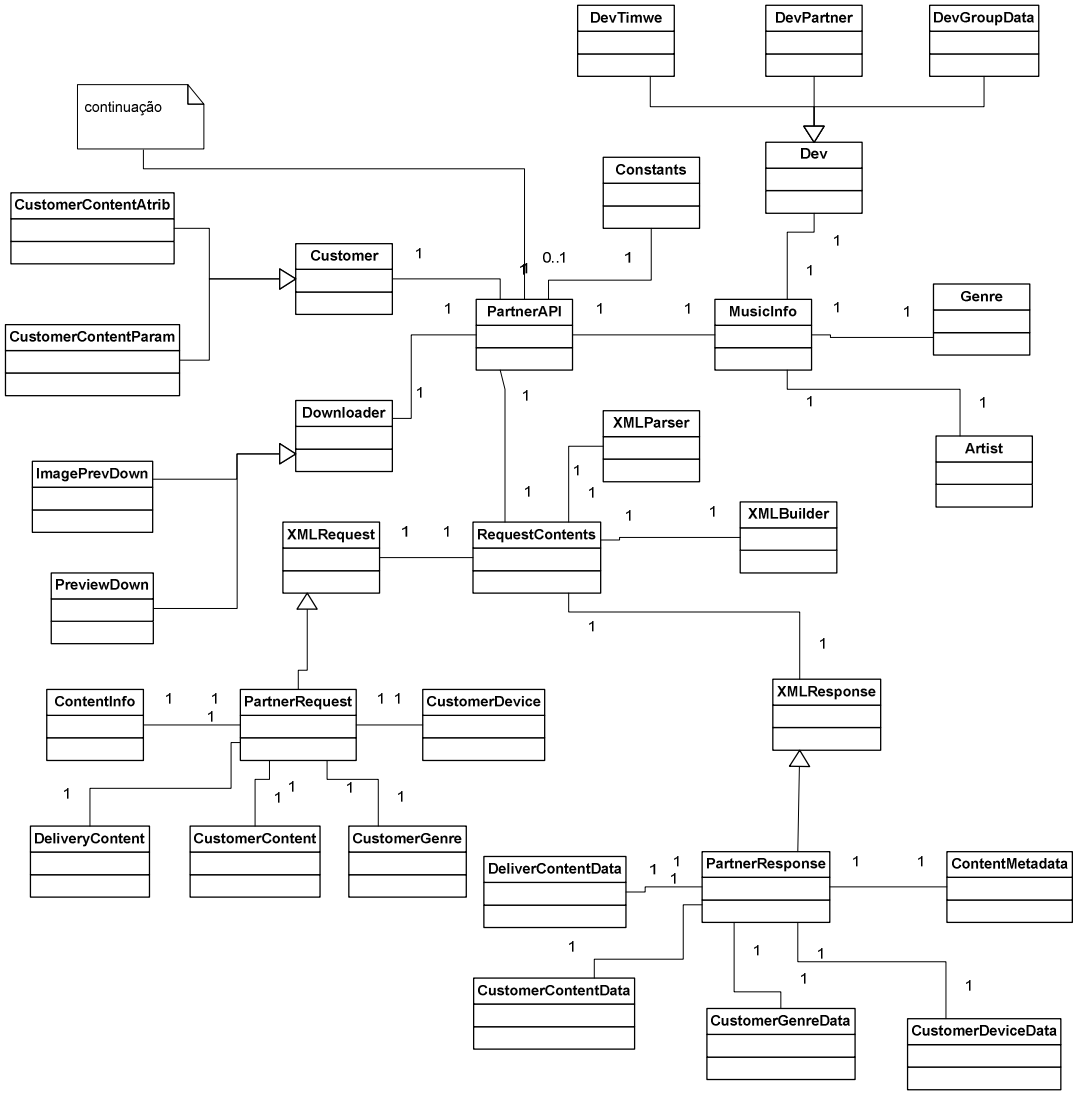
[6] Java API 5.0: http://java.sun.com/j2se/1.5.0/docs/api/

[7] Lewis, John and Loftus, William "Java: Software Solutions, Foundations of Program Design. Fourth edition." Addison-Wesley
http://duke.csc.villanova.edu/jss1/

9. Anexos

9.1. Diagrama de Classes




DevTimwe
DevPartner
DevGroupData
continuação
Dev
Constants
CustomerContentAtrib
Customer
PartnerAPI
MusicInfo
Genre
CustomerContentParam
Downloader
XMLParser
Artist
ImagePrevDown
XMLRequest
RequestContents
XMLBuilder
PreviewDown
XMLResponse
ContentInfo
PartnerRequest
CustomerDevice
DeliveryContent
CustomerContent
CustomerGenre
DeliverContentData
PartnerResponse
ContentMetadata
CustomerContentData
CustomerGenreData
CustomerDeviceData